PROBABILISTIC SCORE PROPAGATION IN INFORMATION RETRIEVAL

BY

AZADEH SHAKERY

B.S., Sharif University of Technology, 2000
M.S., Sharif University of Technology, 2002

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2008

Urbana, Illinois

Doctoral Committee:

Assistant Professor ChengXiang Zhai, Chair
Associate Professor Kevin Chang
Professor Jiawei Han
Research Professor Marianne Winslett

Abstract

Information retrieval techniques deal with different units of information such as terms, topics or documents. There usually exist explicit or implicit link structures between different items of each unit or between items across different units. For example hyperlinks between pages in a hypertext collection are explicit structures, while the links between terms in a co-occurrence network are implicit structures. Many of the traditional information retrieval methods only use the content information of the items for retrieval purposes and overlook the link structures. Those that use the link structures also do not fully exploit the discrimination power of contents as well as all useful link information.

In this thesis, we propose a general probabilistic score propagation framework for combining content and link information, which can fully take advantage of content information and the link structures in a principled way. The basic idea of probabilistic score propagation is to first compute a content-based probability score for each item and then propagate the probabilities through different groups of neighbors. We exploit the content information as a basis to find the content probability score of an item and then use the link structure to define different groups of neighbors to propagate the probabilities through.

We study three applications of this framework for improving retrieval accuracy in three different areas: "Hypertext Retrieval", "Smoothing of Document Language Models" and "Cross-Language Information Retrieval". The experiment results show that the score propagation framework provides a general effective way of exploiting link information along with the content information to improve the retrieval accuracy.

To my parents and Farid,

for their unconditional love and support

Acknowledgments

I would like to express my deepest gratitude to my advisor, ChengXiang Zhai, without whose guidance and encouragement this work would not have been possible. I am also very thankful to my committee members, Jiawei Han, Marianne Winslett and Kevin C. Chang for their constructive suggestions and ideas which helped me to make this dissertation more accurate and complete and inspired many future research possibilities.

I would like to thank all those with whom I have experienced doing research. Special thanks to Bruce Schatz and the BeeSpace group at UIUC for giving me the opportunity to experience doing research in a different area, opening new research possibilities for my future career. Also thanks to my colleagues in the information retrieval group at UIUC whose collaboration has been a great experience for me: Tao Tao, Hui Fang, Xuehua Shen, Jing Jiang, Bin Tan, Qiaozhu Mei, Xuanhui Wang, Alexander Kotov, Maryam Karimzadehgan and Yue Lu.

I am deeply indebted to my parents for all I have accomplished. My heartfelt thanks to my mother who has been my symbol of patience and strength and my father who has been my inspiration all through my studies and life. I am also very grateful to my brother Kaveh for all the joy he has added to my life.

Finally, I wish to express my deepest love to my dearest Farid whose endless love and unceasing support has pulled me through all different situations.

This material is based upon work supported by the National Science Foundation under Grant No. IIS-0347933. Any opinions, findings and conclusions or recomendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation (NSF).

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Information retrieval techniques deal with different units of information such as terms, topics or documents. There usually exist explicit or implicit link structures between different items of each unit or between items across different units. For example hyperlinks between pages in a hypertext collection are explicit structures, while the links between terms in a co-occurrence network are implicit structures. Many of the traditional information retrieval methods only use the content information of the items for retrieval purposes and overlook the link structures. Those that use the link structures also do not fully exploit the discrimination power of contents as well as all useful link information.

For example, many of the typical information retrieval models such as variants of the vector space model [74, 73, 70, 72, 81] and various kinds of logic models and probabilistic models [66, 91, 93, 90, 24, 97, 60] are mainly designed for ranking documents based on content and ignore the link information that may exist between documents. In the area of Web search, many efforts are made to utilize link information for ranking [36, 56, 4, 7, 5, 52, 32, 64, 18, 78, 6, 84, 46, 61]. But none of these methods can fully exploit the discrimination power of contents as well as fully exploit all useful link structures. Despite the importance of link information, the contents of documents are clearly the most *direct* evidence regarding whether a document is relevant to a user's interest. Thus presumably, contents of the documents should be the main basis for ranking them. In this sense, among the proposed link-based ranking methods, only a few ([64, 78, 61]) are close to fully exploiting the content information for ranking. However, they only consider *one* type of *explicit* neighbors and none of them fully take advantage of all the available link information. Thus it is unclear what is the best way to combine content-based scoring and link information.

Beside the explicit link information that exist between information units, one can think of implicit links between units as well. Consider implicit co-occurrence links between terms for example. These links can potentially be useful for improving query representation by bringing in related (additional) terms. However existing work does not fully utilize such link information. For example, the pseudo feedback approach attempts to use co-occurrence information to find related terms to the query terms and to expand the query, but only terms directly co-occurring with the query terms are exploited for expansion in this approach. Content similarity links between documents are another example of implicit link structures. This information can be exploited for improving document representation by smoothing document language models with the content of similar documents. But again the existing work in this direction only considers direct neighbors of each document for smoothing the document language models, not fully utilizing this implicit link information. Therefore it is still unclear how we can best use the implicit link structures available between information units.

In this thesis, we propose a general probabilistic score propagation framework for combining content and link information which can fully take advantage of content information and the link structures in a principled way. The basic idea of probabilistic score propagation is to first compute a content-based probability score for each item and then propagate the probabilities through different groups of neighbors. We exploit the content information as a basis to find the content probability score of each item and then use the link structures to define different groups of neighbors to propagate the probabilities through. After propagation, the model gives us a probabilistic score for each item defined based on a probabilistic surfing model.

Two main characteristics of the proposed framework are the *probabilistic view* on score propagation model and propagation through *multiple groups of neighbors*. Taking a strict probabilistic view of the framework makes the weights of the propagation model more meaningful, providing guidance on how to normalize content scores and how to set other propagation parameters to optimize retrieval accuracy. Experiment results show that appropriate normalization of the weights is often necessary to achieve good performance. Moreover, the proposed framework supports using

multiple types of neighbors for propagation which is shown to outperform the results of using a single type of neighbor.

We study three different applications of the proposed framework to improve retrieval accuracy.

First, we focus on hypertext retrieval as one application and evaluate the framework in ranking search results in a hypertext collection. In this case, the units of information are documents, hyperlinks serve as explicit links and there are various kinds of implicit links, such as co-citation links. We apply the general framework to this document network to come up with a general probabilistic *relevance* propagation framework for hypertext retrieval. We have used the generated probabilistic relevance propagation framework to derive several different models for hypertext retrieval and evaluated the models on two standard TREC Web test collections. The results show that all the derived propagation models can outperform the baseline content-only ranking method over a wide range of parameters, indicating that the relevance propagation framework provides a general, effective and robust way of exploiting link information.

As the second application, we use the framework to design a novel way of smoothing document language models based on propagating term counts probabilistically in the graph of similar documents. In this application, we construct a network of similar documents where the documents are connected based on their content similarity. We apply the general framework to this graph of documents and come up with a probabilistic term count propagation algorithm. In this algorithm, the query term statistics are propagated iteratively in the document network, allowing us to achieve smoothing with *remotely* related documents. Evaluation results on several TREC data sets show that the proposed method outperforms the simple collection-based smoothing method significantly. This method is especially effective in improving precision in top-ranked documents through "filling in" missing query terms in relevant documents, which is presumably most important in practical applications.

In the third application, we use the framework to do cross-language information retrieval where we are given a query in one language and want to retrieve related documents in a second language. For this application, we assume to have very limited linguistic resources, namely comparable

corpora. In this problem, we first construct a word network with words as unit items. The edges between words in the same language indicate mutual information between words and the edges between words in different languages are time correlation edges. We apply the framework to this network to generate a probabilistic score propagation algorithm for cross-language information retrieval. Using this model, we propagate the weights in the word network and construct the query language model in the target language corresponding to the given query. Having the target query language models enable us to retrieve related documents in the target language easily using any typical retrieval method. Evaluation results on TREC-2002 Arabic-English retrieval task show that with the proposed method, we can achieve up to 75.9% of mean average precision, 76.5% of precision at 5 documents and 77.2% of precision at 10 documents compared to the monolingual retrieval performance which is quite promising, since we are using very limited linguistic resources in this application.

The proposed probabilistic score propagation framework is a very general framework which can be exploited in very diverse applications of information retrieval. This framework makes it possible to unify many of the existing link-based (either explicit or implicit) algorithms, allowing us to systematically explore the algorithm space and compare different components of algorithms. Although we study three specific applications of this general framework in this thesis, there are potentially many other applications which can benefit from this general framework.

The rest of the thesis is organized as follows: We introduce the general probabilistic score propagation framework in Chapter 2. We then present the first application, *Probabilistic Relevance Propagation for Hypertext Retrieval* in Chapter 3. In Chapter 4, we will continue with the second application, *Smoothing Document Language Models with Probabilistic Term Count Propagation*. We will present the third application, *Probabilistic Score Propagation for Cross-Language Information Retrieval*, in Chapter 5 and will finally conclude in Chapter 6.

Chapter 2

A General Probabilistic Score Propagation Framework

Information retrieval techniques deal with different units of information, such as terms, topics or documents, with explicit or implicit link structures between them. These link structures are valuable information which can be used along with the content information of units to improve the retrieval accuracy. In this chapter, we propose a general *probabilistic score propagation* framework to combine the score values of different groups of neighbors in the network composed of information units and explicit and/or implicit links between them in a principled way. The basic idea of probabilistic score propagation is to first compute a content-based self probability score for each unit and then propagate the scores through different groups of neighbors in the network.

2.1 Intuition

In different applications in information retrieval, we deal with networks of information units where we have to compute a *quality score* for each node in the network. For example, the World Wide Web is a network of pages with explicit hyperlinks between pages, and we can also think of implicit links between pages, such as co-citation and co-reference links. Co-citation links are between pages that are pointed to by at least one common page and co-reference links are between pages that point to at least one common page. Figure 2.1 shows a typical page p along with different possible neighbors connected through different types of links. In the process of searching for some information, we have to compute a quality score for each page in the network, indicating the relevance of the page to the given query, and sort the pages based on these relevance scores in response to the query.

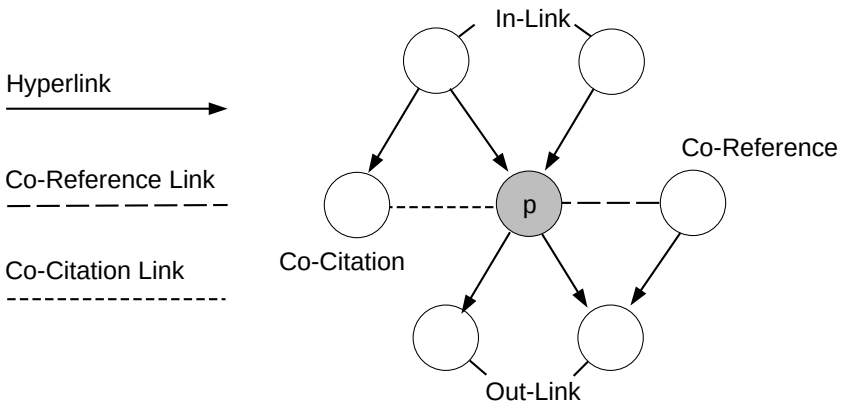


Figure 2.1: A typical page p and possible influential neighbors

Many of the traditional information retrieval methods only use the content information of each node for computing its quality score and overlook the network structure. Those that use the network structure do not take full advantage of the discriminative power of content as well as all useful link information. In the World Wide Web case for example, many traditional search methods use typical retrieval models for ranking which only use the content of each page to compute the quality scores. PageRank [56] is among algorithms that use the link information for this purpose, but it only looks at pages pointing to the page through hyperlinks, ignoring the content information and other implicit link information that exists in the Web network structure. In this area, many efforts are made to utilize link information for ranking, but it is still unclear what is the best way to combine content-based scores and link information and also how to make best use of the implicit link structures.

We observe that the quality score of each node in the network depends both on its content quality and the quality score of different groups of neighbors surrounding the node. For computing the relevance score of each page for example in response to a query, we should not only consider the content of the page, but also the relevance of different groups of neighbors. If the pages that point to the page are highly relevant to the query, there is a high chance that the page itself is a good page on the query. Likewise if the page points to high quality pages, it is itself a high quality

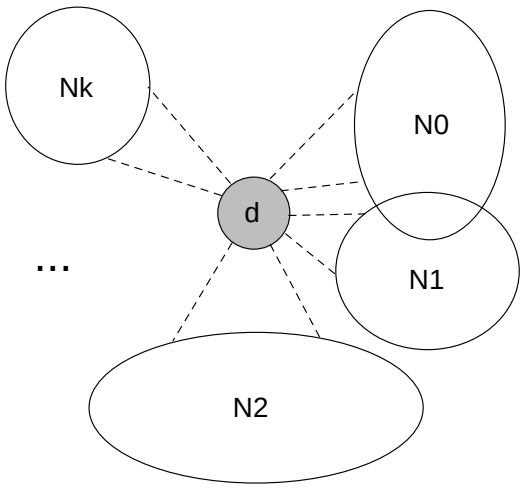


Figure 2.2: A typical node d and its neighbors

page. This is also true for co-citations and co-references. Thus the quality score of a page depends on the quality score of different groups of neighbors. This motivates us to propose a framework which allows different types of neighbors to influence the score of each node.

2.2 Probabilistic Score Propagation Framework

To formalize our intuition, in this section we propose our *general probabilistic score propagation framework* which allows different types of neighbors in the network to influence the quality score of a node. The basic idea of probabilistic score propagation is to first compute a content-based self probability score for each node and then propagate the scores through different groups of neighbors. The framework combines the quality scores of different groups of neighbors in a principled way. Figure 2.2 shows a sample node d surrounded by k different groups of neighbors. The single node itself and the whole set of nodes in the network are possible neighbor sets. Note that the neighbor sets are not necessarily mutually exclusive.

Intuitively, a node has a high score in the network if it is surrounded by high score neighbors. Formally, let x be a node in the network and $N_1, \ldots N_k$ be k different groups of neighbors. We

define the probability score of each node as:

$$p(x) = \sum_{i=1}^{k} \alpha_i \sum_{v \in V} p(v) p_i(v \rightarrow x) \tag{2.1}$$

$$\sum_{i=1}^{k} \alpha_i = 1 \ , \quad \sum_{x \in V} p_i(v \rightarrow x) = 1$$

Here $p(a)$ is the probability score of node a, V is the set of vertices of the network, α_i controls the influence of each group of neighbors on the score of a node and p_i is the influence of a particular node in a neighbor set on the score of the node. $p_i(a \rightarrow b)$ is only positive if b is a neighbor of a, otherwise $p_i(a \rightarrow b) = 0$. Note that we put the restriction $\sum_{x \in V} p_i(v \rightarrow x) = 1$. Intuitively, this means that we have a conditional probability distribution over all the neighbors of v given v, which can be interpreted as being the probability of jumping to a neighbor of v from v. We will use this intuition later in this section as we present the *Random Surfer Model* of this definition. Also note that $\sum_{x \in V} p(x) = 1$. Thus the defined quality scores form a probability distribution.

These probability scores are computed iteratively, updating the score of each node using the updated scores of the neighbors. At each updating step, the score of each node is divided between different groups of neighbors and the score of each group of neighbors is divided between different nodes in the group both in a weighted manner. The score of each node is then updated to the sum of the score portions that the neighbor nodes contribute to the score of the node. The scores are updated iteratively until they converge to a limit. We will show later in the section that the way we propagate the probability scores will guarantee that the scores will converge to a unique probability distribution.

2.2.1 Random Surfer Model

The score definition in equation 2.1 corresponds to the standing probability distribution of a random walk on the network of the information units. Imagine that a random surfer is surfing the information space looking for an information unit related to an information need. At each step,

the surfer being in a node, selects a group of neighbors surrounding the node with probability α_i and jumps to a node in that group with probability $p_i(a \rightarrow b)$ (from the current node a to the destination node b). The surfer keeps doing this iteratively, jumping to different nodes looking for the desired information. The final score of each node is equal to the standing probability of the surfer on the node.

2.2.2 Probabilistic Parameters

In this framework, we identify three groups of probabilistic parameters:

- *Hyper-Score Probability* $p(v)$*:* Defined for each node indicating the probability of visiting the node.
- *Neighbor Set Selection Probability* α*:* Defined for each group of neighbors indicating the probability of choosing a particular type of neighbor set when leaving the current document.
- *Navigation Probability* $p_i(v \rightarrow x)$*:* Defined for each node in a specific neighbor set indicating the probability of visiting a particular node in the chosen neighbor set.

2.2.3 Score Computation

In order to compute the probability scores, for each neighbor set N_i, we construct a matrix M_i where:

$$M_i(m, n) = p_i(v_m \rightarrow v_n)$$

We then compute the probability scores using matrix multiplication: $\vec{P} = M^T \vec{P}$ where $\vec{P}$ is the vector of the probability values and

$$M = \sum_{i=1}^{k} \alpha_i M_i \tag{2.2}$$

The probability values are computed iteratively through matrix multiplications in a very similar way as any of the existing link-based scoring algorithms. Clearly, efficient matrix multiplication

methods can be used to further speed up the scoring. The final scores will be the values of the stationary probability distribution.

2.2.4 Convergence Guarantee

In the proposed framework, we update the probability scores of the nodes iteratively using equation 2.1. We now show that the way we propagate the probability scores will guarantee that the scores will converge to a unique probability distribution, thus we will have a unique final score for each node.

In this framework, we generally include the whole set of vertices of the network as one special group of neighbors. This will ensure reachability to each node in the network. Thus by the Ergodicity theorem for Markov chains [30], we know that the Markov chain defined by the transition matrix M (as defined in 2.2) must have a unique stationary probability distribution.

2.3 Related Work

Markov Chains

In probability theory, a stochastic process has the Markov property if the conditional probability of future states, given the present state, depend only upon the current state. A process with the Markov property is called a Markov chain. Markov chains have been used in many applications in very diverse areas [48, 31, 47]. In this work, we employ Markov chains to perform score propagation in a general way for information retrieval applications. Our proposed probabilistic relevance propagation framework essentially defines a Markov chain with a finite state space, and we cast the score propagation problem as the problem of computing the stationary distribution of the Markov chain over the set of the states. The Markov chain defined in our framework is defined on a finite state space and is ergodic, thus the stationary distribution exists and is unique [62]. While previous work also used Markov chains for ranking purposes (e.g. PageRank [56], topic

specific PageRank [32] and SALSA [42]), our proposed framework provides a more general way of propagating scores for retrieval purposes. See Chapter 3 for more detailed discussion of this line of previous work.

Bayesian Belief Networks

Bayesian belief networks and graphical models [57] are general frameworks for describing joint distributions of a finite number of variables by simplifying the distributions with conditional independence assumptions. These are very general ways of describing any probability distribution where conditional independence assumptions are made. Our framework can be regarded as involving a network of infinite number of variables, more like a dynamic Bayesian network [29]. In most work on Bayesian networks or graphical models, the basic task of probabilistic inference systems is to compute the posterior probability distribution for a set of query variables, given some observed event. Our goal however is to compute the stationary distribution.

Probabilistic Relaxation Labeling Methods

Our proposed propagation framework is similar to the probabilistic relaxation labeling methods [34, 9, 28] in that the relaxation labeling methods also allow the labeling of the neighbors of an object to influence its label. The goal of these algorithms is to assign labels to objects, and they do this probabilistically, i.e. they assign confidence values to the labeling of objects. The confidence values are then updated iteratively based on the configuration of labels of directly interacting objects. But the main concern about these algorithms is that they are not guaranteed to converge. Our framework is superior to these methods in that our probabilistic score propagation framework guarantees the convergence of results.

Spreading Activation Methods

In some sense, this work resembles previous work on spreading activation [10, 71, 16, 17, 75, 22, 59, 76, 44, 15] as both involve propagating values through a network/graph. The main difference,

however, is that in these spreading activation methods, the number of steps for propagating the weights is predefined and is a small value in most of the cases, while our framework is an iterative process which iterates until the scores converge to a limit.

2.4 Applications

The proposed probabilistic relevance propagation framework is a very general framework which can be applied to different applications in different areas. In this thesis, we look into three special applications of this general framework.

In the first application, we use the framework to do hypertext retrieval where we are given a set of documents connected through hyperlinks and a query and the goal is to rank the documents based on their relevance to this query. For this application, we construct a network composed of the documents and different kinds of explicit and implicit links between them and propagate the relevance probability scores of the documents in this network using the proposed framework until they converge to a limit. In response to a given query, we then rank the documents based on these relevance probability scores.

In the second application, we look into a completely different area: smoothing document language models in language modeling approaches to information retrieval. For this application, we construct a similarity graph of documents composed of documents and content similarity links between them and propagate query term statistics in this network using the proposed general probabilistic score propagation framework. Propagating term statistics iteratively in the constructed network allows us to smooth each document with remotely related documents.

Our third application is using the general probabilistic score propagation framework to do cross-language information retrieval, where we are given a query in one language and want to retrieve related documents in another language. The network we construct for this application is composed of terms in the two different languages with mutual information links between terms in the same language and correlation links between terms in different languages. We use the

probabilistic score propagation framework to propagate the term weights in this network and to construct a query language model based on the converged probabilities in the second language. We use this query language model to retrieve documents in this language.

In the rest of the thesis, we will look into these three applications in more detail.

Chapter 3

Probabilistic Relevance Propagation for Hypertext Retrieval

In this chapter, we apply the proposed general probabilistic score propagation framework to a document graph with various kinds of explicit and implicit links, such as hyperlinks and co-citation links, to come up with a general probabilistic *relevance* propagation framework for hypertext retrieval. The generated probabilistic relevance propagation framework can unify most of the existing link-based ranking algorithms, allowing us to systematically compare both the assumptions made in each specific algorithm and different components of the algorithms. It also suggests several interesting new algorithms through different propagation strategies. We will show that the probabilistic relevance propagation framework provides a general, effective and robust way of exploiting link information to improve hypertext search accuracy.

3.1 Introduction

Hypertext Retrieval, the task of searching for information in a hypertext collection, has been around for a while. A key characteristic that distinguishes the search task in a hypertext collection from a traditional retrieval task is the existence of link information in the former one. Although the primary goal of creating links is to guide a user to other parts of the collection, the link information can also be exploited to improve the search accuracy. The existence of this extra information makes it inappropriate to use traditional information retrieval methods, which do the retrieval task based on the content only, to do the search task.

The early works on the hypertext retrieval task were more on the literature side. Some researchers have used bibliographic citation methods to determine relationships among documents

in scientific papers [69, 27, 82]. They have used different citation methods in this direction, namely direct citation, bibliographic coupling - the sharing of one or more references by two documents - and co-citation. Modha and Spangler [50] have proposed a clustering algorithm that clusters hypertext documents using words, out-links and in-links, Chakrabrti et al. [8] have developed a technique called "spectral filtering" for discovering high-quality topical resources in hyperlinked corpora and Ray Larson [40] has applied co-citation analysis methods to the World Wide Web to produce clusterings of the WWW sites that have topical similarities.

Currently with the fast growth and popularity of the World Wide Web, the search task on this huge collection of hypertext data has gained much attention. The problem of hypertext retrieval on the Web has been studied extensively and several link-based ranking methods have been developed to improve retrieval results [36, 56, 4, 7, 5, 52, 32, 64, 18, 78, 6, 84, 46, 83, 100, 105, 35, 88, 2, 61].

Although these algorithms have been shown to improve the performance over some baseline approaches, it remains a challenging research question what is the best way to exploit the content information and the link information to maximize search accuracy. These works appear to have adopted five strategies for combining content and link information: (1) Using the query as a filter to select documents and rank them according to link-based scores (e.g. PageRank [56] and HITS [36]); (2) Computing a weighted combination of topic-specific PageRank scores, where the weights are determined by the query (Topic-sensitive PageRank [32]); (3) Using the query to compute the relevance value of each document and regulating the influence of nodes in HITS using these values (e.g. ARC [7], Bharat and Henzinger[4]); (4) Using the query to compute the relevance value of each document and propagate these values through links (Intelligent Surfer [64], [78], [61]). (5) Using sitemap links to propagate term frequencies ([83], [61]). Unfortunately, none of these combination methods can fully exploit the discrimination power of contents as well as fully exploit all useful link structures. Despite the importance of link information, the contents of documents are clearly the most *direct* evidence regarding whether a document is relevant to a user's interest. Thus presumably, contents of the documents should be the main basis for ranking them. In this sense, among the five strategies, only the last two are close to fully exploiting the

content information to improve ranking. However, the intelligent surfer only considers the in-links of a document, the "relevance propagation" method only considers direct in-links or out-links and the "term propagation" method only considers parent-child links in a sitemap. Each of these methods only considers *one* type of *explicit* neighbors and none of them fully take advantage of all the available link information. Intuitively, all neighbors can be potentially exploited; for example, both out-links and in-links may be useful for ranking as we will show in our experiments. Besides, for the propagation methods, there exist no principled framework to do the propagation. For example, the content scores can be transformed using any monotonic function without affecting the ranking, but such transformation would presumably affect the propagation. How should we transform the scores to achieve the best propagation results?

In this chapter, we show that the proposed general probabilistic score propagation framework, when instantiated on a document graph with various explicit and implicit links, generates a general *probabilistic relevance propagation* framework, offering a general principled way of combining all the link information with content information to improve retrieval accuracy. In this framework, we first compute a content-based relevance probability score for each document using the query, and then propagate the probabilities through different groups of neighbors. We exploit the content information as a basis for finding the probability of the relevance of a document to a query and use the link structures to define different groups of neighbors to propagate the probabilities through.

After propagation, unlike [61], this model gives us a probabilistic score for each document defined based on a probabilistic surfing model. Moreover, this model supports using multiple types of neighbors, which is shown to outperform the results of using a single type of neighbor. On the other hand, the probabilistic interpretation of the model suggests that we should transfer the content-based retrieval scores to probabilities of relevance, which is shown to be beneficial in our experiments. The probabilistic interpretation also provides guidance on how to set various parameters in the propagation model.

We derive several special instances of the general probabilistic relevance propagation framework and show that probabilistic relevance propagation is a very general mechanism that allows

us to recover most of the major existing algorithms as special cases. Moreover, it also naturally suggests several new algorithms that can combine content and link information.

In our experiments, we evaluated several propagation algorithms and the experiment results show that: (1) Using relevance propagation to combine link information and content information for scoring can improve retrieval accuracy over using only content for scoring. (2) Using multiple sets of neighbors for propagation outperforms using a single neighbor set. (3) Using probabilities to control the effect of different groups of neighbors helps. (4) Using probabilities to control the influence of each document in a neighbor set helps.

In the rest of the chapter, we first present our relevance propagation framework and derive several special cases in Section 3.2. We discuss the experiment results in Section 3.3 and present the conclusions and summary on this application in Section 3.4.

3.2 A Probabilistic Relevance Propagation Framework

Given a query, intuitively, a good result document is one whose content is related to the query topic and which is surrounded by other good documents; i.e. located in the center of a subset of the collection relevant to the query. Thus in order to maximize ranking accuracy, we need to consider the relevance of the document to the query as well as the relevance of its neighbors.

We observe that this application is well connected to the proposed general score propagation framework: The units of information are documents, neighbor sets are groups of documents connected through different types of explicit or implicit links, and the propagated scores are the relevance probabilities of the documents to the given query. In this section, we derive a general *probabilistic relevance propagation* framework from the proposed general score propagation framework which allows us to combine relevance values of different groups of neighbors in the document network in a principled way.

3.2.1 Probabilistic Relevance Propagation Framework

In this framework, we allow different types of neighbors to influence the quality score of a document. In-links, Out-links, the single document itself and the whole set of documents are a few examples of potential neighbor sets. We apply the general framework to the document graph with various kinds of links to come up with the general probabilistic relevance propagation framework for hypertext retrieval.

Think of a random surfer surfing the Web looking for documents related to a given query q. At each step, the surfer being in a document, selects a group of neighbors surrounding the document and jumps to a document in that group. The surfer keeps doing this iteratively, jumping to neighbor documents looking for documents relevant to query q. The final score of each document is equal to the stationary probability of the surfer visiting the document.

Formally, the probability of the surfer being in each document is defined as:

$$p(x) = \sum_{i=1}^{k} \alpha_i \sum_{d \in D} p(d) p_i(d \rightarrow x)$$

$$\sum_{i=1}^{k} \alpha_i = 1 \ , \quad \sum_{x \in D} p_i(d \rightarrow x) = 1$$

where D is the set of all documents, α_i indicates the probability of choosing a particular type of neighbor set when leaving the current document and p_i is the probability of visiting a particular page in the chosen neighbor set. Note that $p_i(a \rightarrow b)$ is positive only if b is a neighbor of a, otherwise $p_i(a \rightarrow b) = 0$.

In this framework, as in the general framework, we identify three groups of probabilistic parameters:

- *Hyper-Relevance Probability $p(d)$:* Defined for each document indicating the probability of visiting the document.

- *Neighbor Set Selection Probability α:* Defined for each group of neighbors indicating the

Table 3.1: Probabilistic relevance propagation algorithms - PageRank and its extensions

PageRank [56]	k NB Sets α_is p_is	2 N_0: Set of all docs $\alpha_0 > 0$ const. $P_0(d \to x) = \frac{1}{N}$	 N_I: Set of In-links $\alpha_I > 0$ const. $P_I(d \to x) = \frac{1}{\lvert OUT(d)\rvert}$
Topic-Sensitive PageRank [32]	k NB Sets α_is p_is	2 N_0: Set of all docs $\alpha_0 > 0$ const. $P_0(d \to x) = \begin{cases} \frac{1}{\lvert C_j\rvert} \text{ if } d \text{ in ODPC}^1(c_j) \\ 0 \quad \text{o.w.} \end{cases}$	 N_I: Set of In-links $\alpha_I > 0$ const. $P_I(d \to x) = \frac{1}{\lvert OUT(d)\rvert}$
Intelligent Surfer[64]	k NB Sets α_is p_is	2 N_0: Set of all docs $\alpha_0 > 0$ const. $P_0(d \to x) = \frac{Rel(x)}{\sum_{k \in D} Rel(k)}$	 N_I: Set of In-links $\alpha_I > 0$ const. $P_I(d \to x) = \frac{Rel(x)}{\sum_{d \to k} Rel(k)}$

probability of choosing a particular type of neighbor set when leaving the current document.

- *Navigation Probability* $p_i(d \to x)$*:* Defined for each document in a specific group indicating the probability of visiting a particular page in the chosen neighbor set.

3.2.2 Special Cases

By setting α_is to different values and instantiating p_i's with specific functions, we can easily obtain many special cases of our general relevance propagation framework. In particular, the framework can recover most existing link-based ranking algorithms. Tables 3.1 and 3.2 show two groups of relevance propagation algorithms which are covered by our general framework.

As can be seen from the tables, PageRank and its extensions are special cases of the framework. The HITS algorithm is not directly a special case, since it does not satisfy the probability property. But with minor changes, i.e. normalization of the weights, it will be a special case. In Table 3.2,

Table 3.2: Probabilistic relevance propagation algorithms - HITS and its extensions

Normalized HITS[36]	Authorities	k	1
		NB Sets	N_{CC}: Set of Co-Citations
		α_is	$\alpha_{CC} = 1$
		p_is	$P_{CC}(d \to x) \propto \begin{cases} \text{\|IN}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Parents} & \text{o.w.} \end{cases}$
	Hubs	k	1
		NB Sets	N_{CR}: Set of Co-References
		α_is	$\alpha_{CR} = 1$
		p_is	$P_{CR}(d \to x) \propto \begin{cases} \text{\|OUT}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Children} & \text{o.w.} \end{cases}$
Normalized Weighted HITS[4]	Authorities	k	1
		NB Sets	N_{CC}: Set of Co-Citations
		α_is	$\alpha_{CC} = 1$
		p_is	$P_{CC}(d \to x) \propto Rel(x) \times \begin{cases} \text{\|IN}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Parents} & \text{o.w.} \end{cases}$
	Hubs	k	1
		NB Sets	N_{CR}: Set of Co-References
		α_is	$\alpha_{CR} = 1$
		p_is	$P_{CR}(d \to x) \propto Rel(x) \times \begin{cases} \text{\|OUT}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Children} & \text{o.w.} \end{cases}$
Normalized ARC[7]	Authorities	k	1
		NB Sets	N_{CC}: Set of Co-Citations
		α_is	$\alpha_{CC} = 1$
		p_is	$P_{CC}(d \to x) \propto Rel(anchor(x)) \times \begin{cases} \text{\|IN}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Parents} & \text{o.w.} \end{cases}$
	Hubs	k	1
		NB Sets	N_{CR}: Set of Co-References
		α_is	$\alpha_{CR} = 1$
		p_is	$P_{CR}(d \to x) \propto Rel(anchor(x)) \times \begin{cases} \text{\|OUT}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Children} & \text{o.w.} \end{cases}$
Normalized Randomized HITS[52]	Authorities	k	2
		NB Sets	N_0: Set of all docs N_{CC}: Set of Co-Citations
		α_is	$\alpha_0 > 0$ const. $\alpha_{CC} = 1$
		p_is	$P_0(d \to x) = \frac{1}{N}$ $P_{CC}(d \to x) \propto \begin{cases} \text{\|IN}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Parents} & \text{o.w.} \end{cases}$
	Hubs	k	2
		NB Sets	N_0: Set of all docs N_{CR}: Set of Co-References
		α_is	$\alpha_0 > 0$ const. $\alpha_{CR} = 1$
		p_is	$P_0(d \to x) = \frac{1}{N}$ $P_{CR}(d \to x) \propto \begin{cases} \text{\|OUT}(d)\text{\|} & \text{if } d = x \\ \#\text{Common Children} & \text{o.w.} \end{cases}$

we include the normalized version of HITS as well as the normalized version of its extensions.

3.2.3 Parameter Estimation

In this framework, we have identified three groups of probabilistic parameters: content relevance probabilities, neighbor set selection probabilities and navigation probabilities. The content relevance probability of a document can be estimated based on its relevance score given by any content-based retrieval method. Neighbor set selection probabilities and navigation probabilities can either set to be uniform or estimated based on content-based relevance scores. In this section, we show how to estimate the parameters. We compare different estimation methods in the following section.

Content Relevance Probabilities

In our probabilistic framework we should convert the original content scores to probabilities. The specific conversion method is inevitably dependent on the specific content scoring method, but with some training data, we may use techniques such as logistic regression [65] to do the conversion. If the original retrieval model is a probabilistic model, we have some natural analytical way to transform the scores. As an example of this transformation, here we show how we compute relevance probabilities from Okapi scores and from Language Model(LM) scores.

1. Okapi

   Having the Okapi scores, our goal is to normalize the scores to find the *relevance probabilities* of the documents to the query. We use logistic regression to normalize the scores [65]:

   The Okapi score is $a*X+b$, where X is the log odds of relevance, i.e., $\log(p(rel)/(1-p(rel))$. So, to recover the probability $p(rel)$, we have $p(rel) = \exp(X)/(1+\exp(X))$. Given a score s, we have $s = aX + b$, or $X = (s-b)/a$. Thus, the normalization formula should be $p(rel) = \exp((s-b)/a)/(1+\exp((s-b)/a))$.

In order to set a and b, we assume that the minimum score *min* corresponds to a very small probability δ. We also assume that the maximum score *max* corresponds to $p(rel) = \Delta$. Solving these equations will give us values for a and b:

$$a = \frac{min - max}{\log(\frac{\delta}{1-\delta}) - \log(\frac{\Delta}{1-\Delta})}$$

$$b = \frac{max \times \log\frac{\delta}{1-\delta} - min \times \log\frac{\Delta}{1-\Delta}}{\log\frac{\delta}{1-\delta} - \log\frac{\Delta}{1-\Delta}}$$

2. Language Modeling Approach

   In the language modeling approach, we score a document D with respect to a query Q by $s = \log p(Q|D)$ [101]. Thus we can do an exponential transformation to recover the probabilities of relevance. That is, $p(rel) \propto p(Q|D)p(D) \propto exp(s)$ (assuming uniform p(D)).

Neighbor Set Selection Probabilities

The easiest way to estimate neighbor set selection probabilities is uniform estimation, counting all the neighbor sets to be equal, i.e. $\alpha_i = \frac{1}{k}$.

But obviously this is not the best we can do. Our framework suggests to use relevance scores for Neighbor set probability estimation. We get our intuition for defining neighbor set selection probabilities from the surfer model. In the surfer model, in each step, the surfer should decide on the neighbor set it wants to jump to. Intuitively, the surfer will select the neighbor set based on the average relevance of the documents in the neighbor set, the higher the average relevance, the more probable the surfer will select that group. Using this intuition, we set α_i using:

$$\alpha_i \propto \frac{1}{|N_i|} \sum_{X \in N_i} rel(X)\ , \qquad \sum \alpha_i = 1$$

Navigation Probabilities

Like neighbor set selection probabilities, the navigation probabilities are most easily estimated through uniform estimation. But intuitively, estimating the probabilities using relevance values should give better results.

We define navigation probabilities based on the content relevance probabilities of target pages. The higher the probability of the relevance of the target page, the higher the probability of navigating the link: $p(d \rightarrow x) \propto p(x)$.

3.2.4 Summary

The derived framework provides a general probabilistic interpretation of relevance-based propagation through multiple sets of neighbors. It can unify most existing link-based ranking algorithms, making it possible to compare the assumptions made in each specific algorithm. It also makes it possible to systematically explore the algorithm space and compare different components of algorithms. Moreover, taking a strict probabilistic view of propagation provides guidance on how to normalize content scores and how to set other propagation parameters to optimize retrieval accuracy, as will be shown later.

3.3 Comparison of Relevance Propagation Algorithms

We have done some experiments to evaluate the performance of our proposed models. In this section, we present our experiment results.

3.3.1 Experiment Design

Data Set and Baseline Methods

As the data set, we used the ".GOV" test collection, which is an 18 gigabyte, 1.25 million document 2002 partial crawl of the .gov domain used in TREC-2002, TREC-2003 and TREC-2004

experiments for topic distillation [11, 12, 13]. We used two sets of queries in our experiments: (1) 50 topic distillation topics created by NIST for TREC-2003 and (2) 75 topic distillation topics created by NIST for TREC-2004. The topics are keyword queries for which key resources exist within the .GOV collection.

An important advantage of using this data set is that it is created carefully for the purpose of evaluating Web retrieval algorithms with a significant number of judgments available for quantitatively comparing different methods.

In our experiments, we used two baseline methods: Okapi and Language Modeling approach. Since our exploration is orthogonal to the use of anchor text and many other heuristics which are known to improve the performance, we preferred not to enter these heuristics in our baseline. Despite this, we already have a very strong baseline compared to the reported results in TREC-2003 [12] and TREC-2004 [13]. We expect the performance to be further improved when we use other heuristics on top of our method.

Neighbor Sets

In our experiments, we compare the performance of using two types of neighbors: The set of documents which have links to the document(IN) and the set of documents which are linked from the document(OUT). There also exist a universal neighbor set N_0 which contains all the documents in the collection. Selecting this universal neighbor set to jump to is equivalent to jumping to a random page.

Content Relevance Probabilities

The probabilistic relevance propagation framework allows us to use any content-based retrieval algorithm from which we can compute the relevance probabilities. In our experiments, we try two baseline methods: Okapi and language modeling approach and compute the relevance probabilities from these relevance scores.

Neighbor Set Selection and Navigation Probabilities

As mentioned earlier, α_is are the parameters which indicate the probability of choosing a particular type of neighbor when leaving the current document. In our experiments, we follow one of the two approaches: either manually set α_i to different values from 0 to 1 or automatically set α_i using neighbor set selection probabilities.

Navigation probabilities on the other hand indicate the probability of visiting a particular page in a group. In our experiments, we use two different estimations of these probabilities: Uniform estimation ("Uni") and relevance based estimation ("Wt").

3.3.2 Result Analysis

Effectiveness of Exploiting Link Information

The first research question we want to answer is whether applying probabilistic relevance propagation on top of a content-based retrieval method would improve the performance. Most existing studies of link-based scoring algorithms focus on comparing different link-based algorithms without comparing link-based algorithms with scoring using only contents. The Web Track of TREC has seen some evaluation of effectiveness of exploiting link information to improve content-based scoring, but the results are not quite conclusive due to the many uncontrolled factors.

To answer the first research question, we compare the performance of using two types of neighbors: in-links and out-links. (Note that we also have the universal neighbor set). We will have:

$$\begin{aligned} p(x) &= \alpha_0 \Sigma_{d \in D} p(d) p_0(d \rightarrow x) \\ &+ \alpha_I \Sigma_{d \in IN} p(d) p_I(d \rightarrow x) \\ &+ \alpha_O \Sigma_{d \in OUT} p(d) p_O(d \rightarrow x) \end{aligned}$$

where α_0 is the probability of randomly jumping to a page, α_I is the probability of jumping to an in-link and α_O is the probability of jumping to an out-link. Jumping probabilities can either be

Table 3.3: Combining link and content information - Okapi baseline

Method	TREC-2003			
	Prec@10	Impr.	MAP	Impr.
Baseline	0.108	-	0.121	-
Uni-IN	0.118	9.3%	0.145	20%
Wt-IN	0.128	18.5%	0.144	19%
Uni-OUT	0.118	9.3%	0.151	24.8%
Wt-OUT	0.122	13%	0.163	34.7%
Uni-IN Uni-OUT	0.126	16.7%	0.168	38.8%
Uni-IN Wt-OUT	0.132	22.2%	0.166	37.2%
Wt-IN Uni-OUT	0.138	27.8%	0.173	43%
Wt-IN Wt-OUT	0.138	27.8%	0.179	47.9%
	TREC-2004			
	Prec@10	Impr.	MAP	Impr.
Baseline	0.129	-	0.093	-
Uni-IN	0.18	39.5%	0.125	34.4%
Wt-IN	0.181	40.3%	0.125	34.4%
Uni-OUT	0.156	20.9%	0.112	20.4%
Wt-OUT	0.157	21.7%	0.113	21.5%
Uni-IN Uni-OUT	0.179	38.8%	0.124	33.3%
Uni-IN Wt-OUT	0.184	42.6%	0.127	36.6%
Wt-IN Uni-OUT	0.18	39.5%	0.125	34.4%
Wt-IN Wt-OUT	0.188	45.7%	0.127	36.6%

uniform (considering all the members to be equal) or weighted based on relevance probabilities. We also consider the combination of the two types of neighbors. This gives us eight combinations, which we compare with the content-only baseline in Tables 3.3 and 3.4. We use precision at 10 documents (Prec@10) and Mean Average Precision (MAP) for comparison. The shown results are the best performances achieved by these methods through tuning the parameter α manually in the probabilistic relevance propagation model; we will analyze the sensitivity later.

From Tables 3.3 and 3.4, we can make the following observations:

1. On both query sets, both types of neighbors can outperform the baseline significantly.

2. Weighted propagation of probabilities outperforms uniform propagation.

3. The combination of different types of neighbors outperforms using any single neighbor set.

Table 3.4: Combining link and content information - LM baseline

Method	TREC-2003			
	Prec@10	Impr.	MAP	Impr.
Baseline	0.092	-	0.099	-
Uni-IN	0.118	28.3%	0.135	36.4%
Wt-IN	0.118	28.3%	0.142	43.4%
Uni-OUT	0.106	15.2%	0.129	30.3%
Wt-OUT	0.11	19.6%	0.135	36.3%
Uni-IN Uni-OUT	0.118	28.3%	0.150	51.5%
Uni-IN Wt-OUT	0.122	32.6%	0.142	43.4%
Wt-IN Uni-OUT	0.126	37%	0.145	46.5%
Wt-IN Wt-OUT	0.128	39.1%	0.144	45.5%
	TREC-2004			
	Prec@10	Impr.	MAP	Impr.
Baseline	0.129	-	0.095	-
Uni-IN	0.165	27.9%	0.113	18.9%
Wt-IN	0.167	29.5%	0.115	21.1%
Uni-OUT	0.141	9.3%	0.107	12.6%
Wt-OUT	0.144	11.6%	0.11	15.8%
Uni-IN Uni-OUT	0.16	24%	0.115	20.8%
Uni-IN Wt-OUT	0.163	26.4%	0.116	21.1%
Wt-IN Uni-OUT	0.164	27.1%	0.117	23.2%
Wt-IN Wt-OUT	0.167	29.5%	0.119	25.3%

4. We get significant improvement using both Okapi and LM baselines.

Overall, we see that the probabilistic relevance propagation framework is reasonable and all these specific derived algorithms can help improve search results.

Effectiveness of Combining Different Groups of Neighbors

In Tables 3.5 and 3.6, we compare the results of using only one type of neighbor with the results when we consider multiple groups of neighbors. We did a Wilcoxon signed rank test to see if the improvement on mean average precision is statistically significant. In these tables we compare the best results for each type of neighbor. Statistically significant improvements are distinguished by a star($*$).

Table 3.5: Using multiple neighbors versus a single neighbor set-TREC-2003

Okapi Baseline			
Multi Neighbor Sets	Single Neighbor Set		Improvement
Uni-IN Uni-OUT 0.168	Uni-IN	0.145	15.9% *
	Uni-OUT	0.151	11.3%
Uni-IN Wt-OUT 0.166	Uni-IN	0.145	14.5% *
	Wt-OUT	0.163	1.8%
Wt-IN Uni-OUT 0.173	Wt-IN	0.144	20.1% *
	Uni-OUT	0.151	14.6%
Wt-IN Wt-OUT 0.179	Wt-IN	0.144	24.3% *
	Wt-OUT	0.163	9.8%
LM Baseline			
Multi Neighbor Sets	Single Neighbor Set		Improvement
Uni-IN Uni-OUT 0.150	Uni-IN	0.135	11.1% *
	Uni-OUT	0.129	16.3%
Uni-IN Wt-OUT 0.142	Uni-IN	0.135	5.2% *
	Wt-OUT	0.135	5.2%
Wt-IN Uni-OUT 0.145	Wt-IN	0.142	2.1%
	Uni-OUT	0.129	12.4% *
Wt-IN Wt-OUT 0.144	Wt-IN	0.142	1.4%
	Wt-OUT	0.135	6.7% *

As the tables show, combining different groups of neighbors improves the performance over using a single set of neighbors. Potentially, we can improve the performance by adding new types

Table 3.6: Using multiple neighbors versus a single neighbor set-TREC-2004

Okapi Baseline			
Multi Neighbor Sets	Single Neighbor Set		Improvement
Uni-IN Uni-OUT	Uni-IN	0.125	-
0.124	Uni-OUT	0.112	10.7% *
Uni-IN Wt-OUT	Uni-IN	0.125	0.8%
0.126	Wt-OUT	0.113	11.5%
Wt-IN Uni-OUT	Wt-IN	0.125	-
0.125	Uni-OUT	0.112	11.6% *
Wt-IN Wt-OUT	Wt-IN	0.125	1.6%
0.127	Wt-OUT	0.113	12.4% *
LM Baseline			
Multi Neighbor Sets	Single Neighbor Set		Improvement
Uni-IN Uni-OUT	Uni-IN	0.113	1.8%
0.115	Uni-OUT	0.107	7.5% *
Uni-IN Wt-OUT	Uni-IN	0.113	2.7%
0.116	Wt-OUT	0.11	5.5% *
Wt-IN Uni-OUT	Wt-IN	0.115	1.7%
0.117	Uni-OUT	0.107	9.3% *
Wt-IN Wt-OUT	Wt-IN	0.115	3.5%
0.119	Wt-OUT	0.11	8.1% *

of neighbors, e.g. co-citations (documents which have at least one common parent with the document) and co-references (documents which have at least one common child with the document).

Content Score Transformation

The probabilistic framework suggests that we should convert the original content scores to probabilities. In our experiments, we use Okapi and LM methods as our baseline and transform the scores to probabilities using logistic regression and exponential transformation respectively.

Figures 3.1, 3.2, 3.3 and 3.4 compare the performance of probabilistic transformation with the performance of the original raw score propagation as done in all the previous work. As can be seen, the performance is much better when we use probabilistic transformation.

Comparison of Estimation Methods

- Relevance-Based Estimate of α Improves over Uniform Estimate.

  In one set of experiments, we tried to set αs automatically based on the average relevance values of neighbors. Table 3.7 compares the results of relevance-based estimation of α with uniform estimation. As the table shows, in most of the cases relevance-based estimation gives better results. Note that these results are completely automatic; i.e. we do not have to tune any parameters. Thus these improvements are very encouraging. These results also confirm that using multiple neighbor sets improves over using just a single neighbor set.

Table 3.7: Effectiveness of neighbor set selection probability estimation (α)

Method	Uniform Estimate		Relevance Estimate	
	Prec@10	**MAP**	Prec@10	**MAP**
Baseline	0.108	**0.1206**	0.108	**0.1206**
Uni-IN	0.104	**0.1191**	0.114	**0.1438**
Wt-IN	0.114	**0.1434**	0.124	**0.1496**
Uni-OUT	0.114	**0.1397**	0.114	**0.1563**
Wt-OUT	0.116	**0.159**	0.118	**0.1556**
Uni-IN Uni-OUT	0.116	**0.1352**	0.118	**0.1586**
Uni-IN Wt-OUT	0.124	**0.1578**	0.122	**0.16**
Wt-IN Uni-OUT	0.124	**0.1677**	0.13	**0.1699**
Wt-IN Wt-OUT	0.128	**0.175**	0.138	**0.1694**

- Relevance-Based Estimate of $p_i(d \rightarrow x)$ Improves over Uniform Estimate.

  In Table 3.8, we compare the results of uniformly setting the navigation weights versus estimating them based on relevance scores. As the table shows, relevance based estimation improves the performance in most of the cases.

Sensitivity Analysis

We have so far only looked at the best performance using each method. We now turn to the question about how sensitive each method is to the setting of the parameter α, which controls the amount of influence from the neighbors. To answer this research question, we compute an "optimal range" of

Table 3.8: Navigation probability estimation - TREC-2003

Okapi Baseline				
Neighbor Set	Uniform Estimate		Relevance Estimate (Impr.)	
	Prec@10	**MAP**	Prec@10	**MAP**
IN	0.118	**0.145**	0.128(8.5%)	**0.144(-)**
OUT	0.118	**0.151**	0.122(3.4%)	**0.163(8.1%)**
IN & OUT	0.126	**0.168**	0.138(9.5%)	**0.179(6.5%)**
LM Baseline				
Neighbor Set	Uniform Estimate		Relevance Estimate (Impr.)	
	Prec@10	**MAP**	Prec@10	**MAP**
IN	0.118	**0.135**	0.118(-)	**0.142(5.2%)**
OUT	0.106	**0.129**	0.11(3.8%)	**0.135(4.7%)**
IN & OUT	0.118	**0.150**	0.128(8.5%)	**0.144(-)**

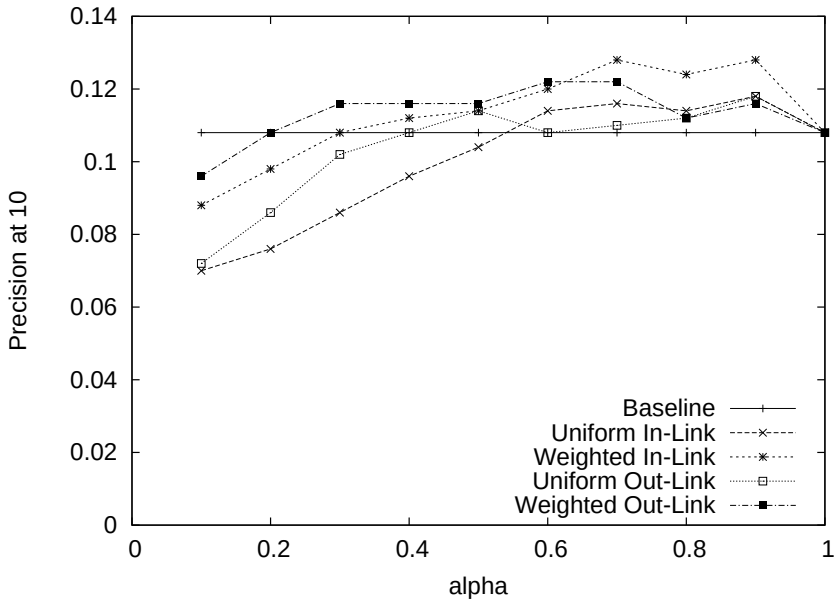
parameter values for each method, which is defined as the interval of parameter values for which a method outperforms the baseline. Table 3.9 shows the optimal ranges for four of our algorithms.

Table 3.9: Ranges of α values for improving baselines

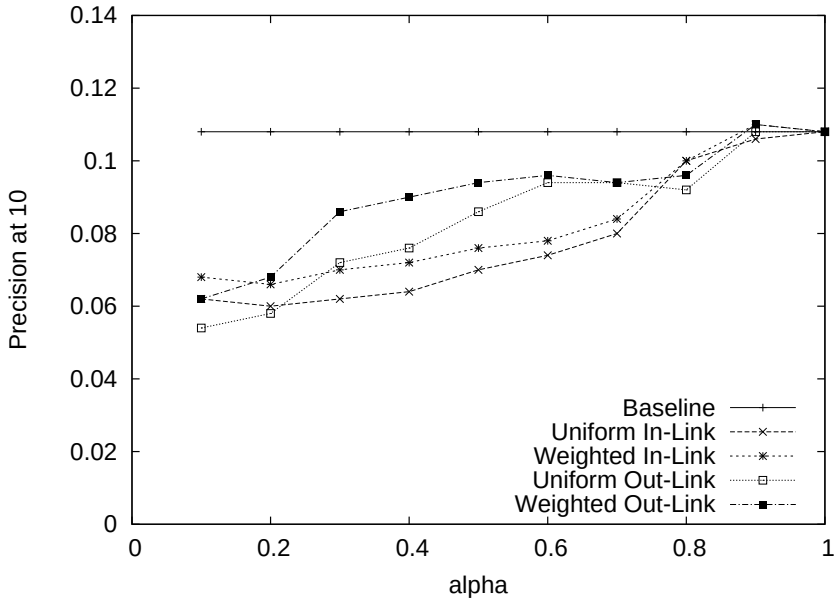
Method	Okapi Baseline			
	TREC-2003		TREC-2004	
	Prec @ 10	**MAP**	Prec @ 10	**MAP**
Uni-IN	[0.6, 0.9]	**[0.6, 0.9]**	[0.3, 0.9]	**[0.3, 0.9]**
Wt-IN	[0.3, 0.9]	**[0.3, 0.9]**	[0.3, 0.9]	**[0.2, 0.9]**
Uni-OUT	[0.4, 0.9]	**[0.4, 0.9]**	[0.6, 0.9]	**[0.4, 0.9]**
Wt-OUT	[0.2, 0.9]	**[0.2, 0.9]**	[0.3, 0.9]	**[0.2, 0.9]**
Method	LM Baseline			
	TREC-2003		TREC-2004	
	Prec @ 10	**MAP**	Prec @ 10	**MAP**
Uni-IN	[0.5, 0.9]	**[0.6, 0.9]**	[0.6, 0.9]	**[0.6, 0.9]**
Wt-IN	[0.3, 0.9]	**[0.2, 0.9]**	[0.3, 0.9]	**[0.4, 0.9]**
Uni-OUT	[0.6, 0.9]	**[0.3, 0.9]**	[0.7, 0.9]	**[0.6, 0.9]**
Wt-OUT	[0.5, 0.9]	**[0.1, 0.9]**	[0.6, 0.9]	**[0.3, 0.9]**

We see that, in general, the optimal range is wide for most methods, indicating that exploiting these groups of neighbors for relevance propagation is useful. The uniform methods are generally more sensitive to the setting of α, which indicates that weighted methods are more robust.

In Figures 3.1, 3.2, 3.3 and 3.4, we show the complete picture of the sensitivity of these meth-

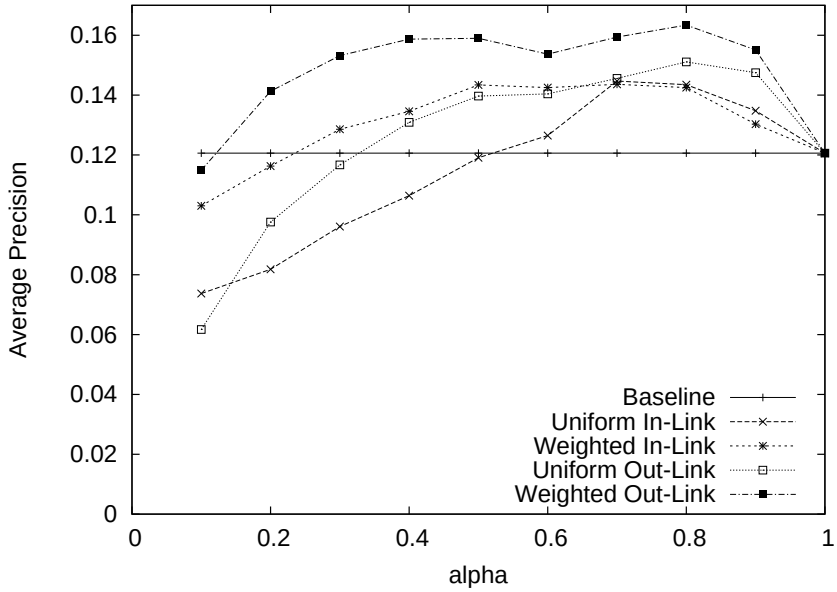


(a) Results using relevance probabilities

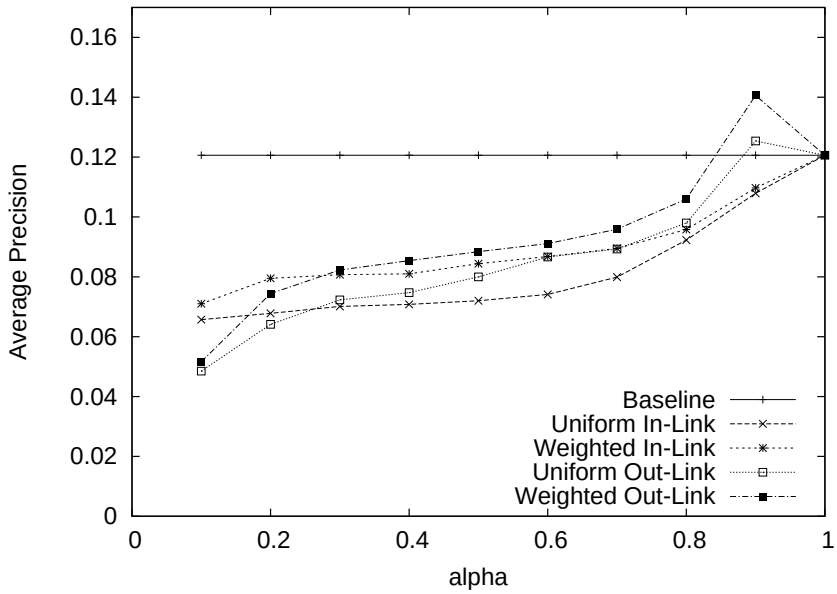


(b) Results using raw content scores

Figure 3.1: TREC-2003 Precision at 10 documents - Okapi baseline



(a) Results using relevance probabilities



(b) Results using raw content scores

Figure 3.2: TREC-2003 Mean Average Precision - Okapi baseline

ods.

3.4 Summary

In this chapter, we applied the proposed general framework to a document network and derived a general *probabilistic relevance propagation* framework for hypertext retrieval for combining content and link information in a principled manner to fully take advantage of query-based content scoring and link structures. The framework can unify most existing link-based ranking algorithms and can also suggest several interesting new algorithms through different propagation strategies.

Following the probabilistic relevance propagation framework, we systematically compared eight specific relevance propagation models on two TREC test collections for Web retrieval. Our results show that all the eight relevance propagation models that we tested can outperform the baseline content only ranking method for a wide range of parameter values, indicating that the relevance propagation framework provides a general, effective and robust way of exploiting link information to improve hypertext search accuracy.

While the previous work all uses just one type of neighbor for propagation, we have shown that using multiple neighbor sets outperforms using just one type of neighbors significantly. We have also shown that taking a probabilistic view of propagation provides guidance on setting propagation parameters, that using content scores to estimate the probabilities of relevance improves the performance and that relevance based estimation of the parameters helps us improve the results.

There are several interesting directions for further research:

1. Our framework naturally accommodates the use of anchor text through estimating navigation parameters based on anchor text. It is interesting to see how this estimation compares with our current estimation methods.

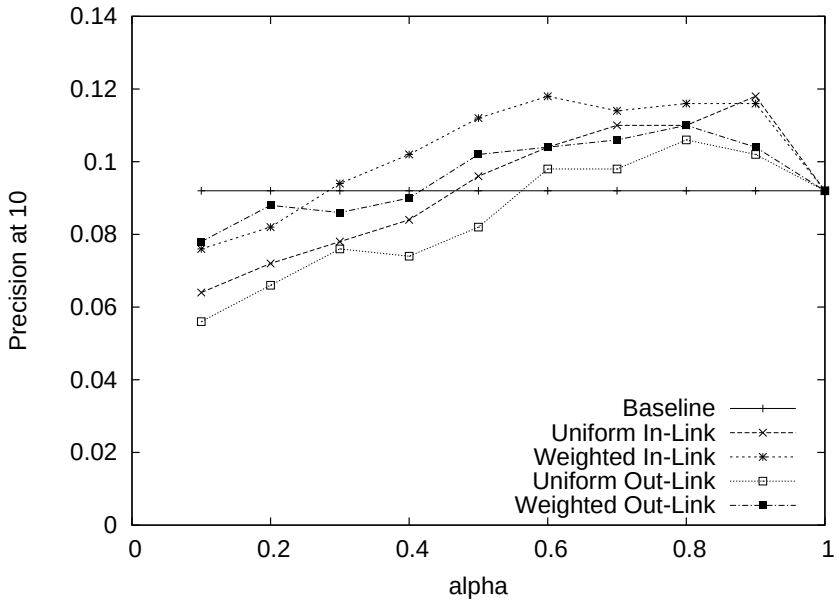
2. We have shown that in-links and out-links are useful for relevance propagation and can outperform the "content only" baseline. It would be interesting to try other kinds of neighbors,

e.g. co-citations and co-references to see if they can further improve the performance.

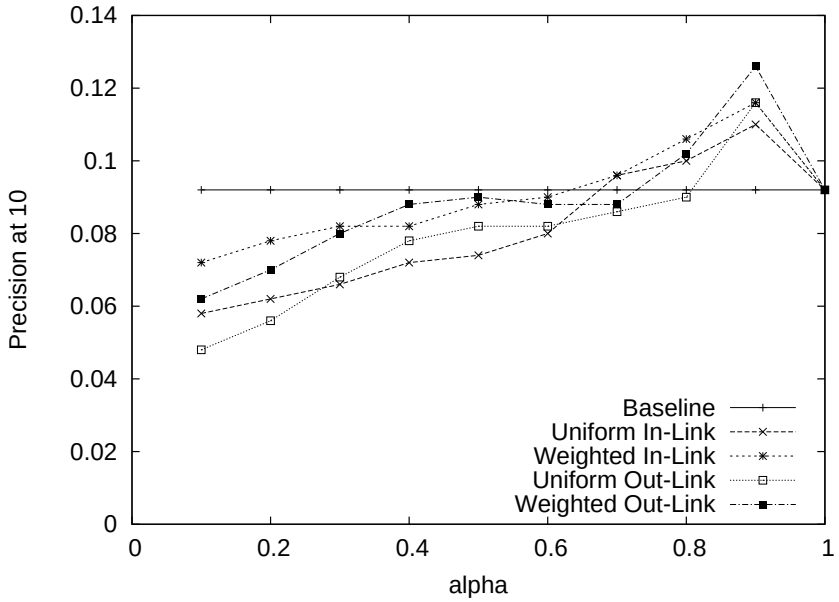
3. Other than the neighbor sets derived from the explicit link structure of the Web, we can also define other types of neighbors. In general, the framework allows us to define any set of documents with a specific characteristic as a neighbor set. As an example, we can define the set of pages with similar contents as a neighbor set. It is interesting to see if exploiting these types of neighbors can further improve the retrieval accuracy.

4. We currently define the neighbor sets based on the links between nodes. For example in a hypertext collection, all the nodes pointing to a specific node are grouped together to construct the ”in-links” neighbor set. We can further refine the neighbor sets, dividing a neighbor set to smaller subgroups each having a specific property and treat each subgroup as a different type of neighbors. This refinement allows us to treat each subgroup differently by giving them different weights.

5. The probabilistic relevance propagation framework is a general hypertext retrieval framework that can be applicable to any hypertext retrieval environment. For example, we may apply the algorithms we studied here to literature search where the links represent citations.

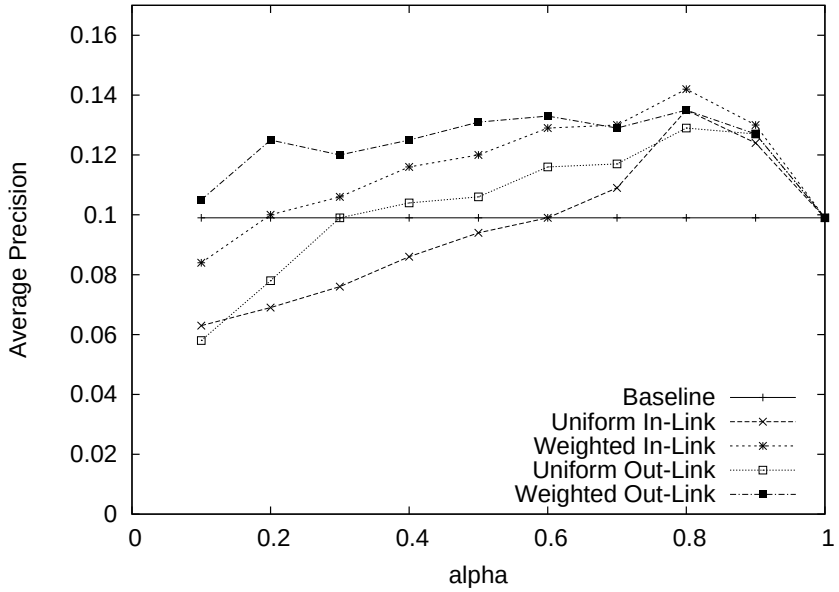


(a) Results using relevance probabilities

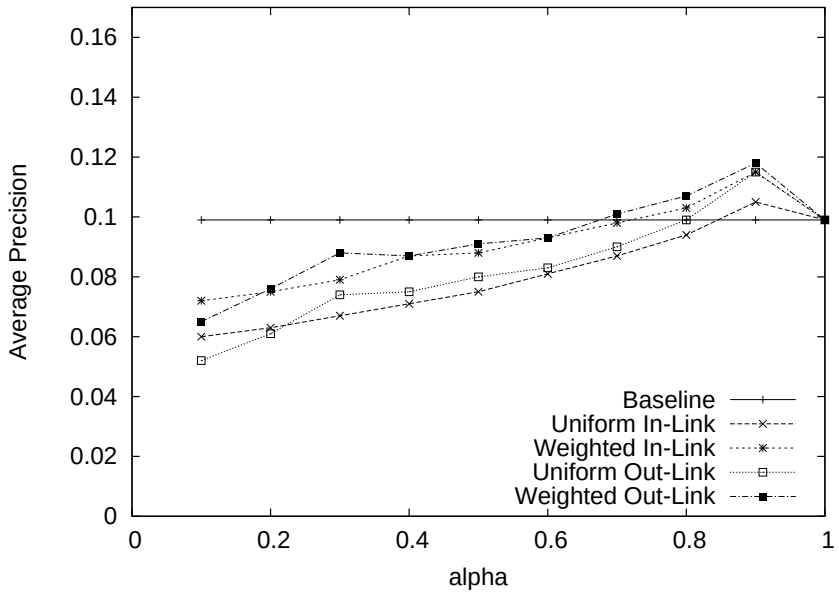


(b) Results using raw content scores

Figure 3.3: TREC-2003 Precision at 10 documents - LM baseline



(a) Results using relevance probabilities



(b) Results using raw content scores

Figure 3.4: TREC-2003 Mean Average Precision - LM baseline

Chapter 4

Smoothing Document Language Models with Probabilistic Term Count Propagation

This chapter focuses on smoothing document language models in language modeling approaches to information retrieval. We will apply the proposed general probabilistic score propagation framework to a graph of documents with implicit generation links to come up with a probabilistic term count propagation algorithm. Generation links can be thought of as automatically generated citation links between related documents. The generated probabilistic term count propagation algorithm allows us to iteratively propagate term count statistics in the graph of similar documents to achieve smoothing with *remotely* related documents. We will show that this method of smoothing significantly outperforms the simple collection-based smoothing method. Compared with other smoothing methods that exploit local corpus structures, this method is especially effective in improving precision in top-ranked documents through "filling in" missing query terms in relevant documents, which is attractive since most users only pay attention to the top-ranked documents in search engine applications.

4.1 Introduction

In language modeling approaches to information retrieval, we often score a document based on the likelihood of a query according to a document language model [60, 102] or the KL-divergence between a query language model and a document language model [39, 102]. In any case, a basic task is to estimate a document language model. In [102], it is shown that accurate estimation of document language models is quite critical for improving retrieval performance, and in particular, how to smooth a document language model can significantly affect the retrieval precision.

Traditional smoothing methods mainly use the global collection information for smoothing [60, 49, 33, 103]. These methods generally do a linear interpolation of the maximum likelihood estimate of the model and a reference language model estimated using the whole collection:

$$p_{smooth}(w|d) = (1 - \lambda)p_{ML}(w|d) + \lambda p(w|\mathcal{C})$$

where $p_{ML}(w|d)$ is the maximum likelihood estimate of the model, $p(w|\mathcal{C})$ is the collection language model and coefficient λ controls the influence of each model. Thus these methods use the probabilities computed based on the whole document collection for smoothing.

Recently there has been some research on using local corpus structure for smoothing purposes [43, 37, 85]. These methods use local corpus information instead of global information for smoothing with the intuition that local structure provides more focused information about the document. These methods also use a simple interpolation of the maximum likelihood estimate of the model and the local surrounding model for smoothing:

$$p_{smooth}(w|d) = (1 - \lambda)p_{ML}(w|d) + \lambda p(w|c)$$

where $p(w|c)$ is the local surrounding model. What all these smoothing algorithms do in common is a simple *one step* interpolation of the model derived from the individual document and the model of the surrounding documents. Note that the surrounding can potentially include all the documents, based on how we define the surrounding.

In this chapter, we propose a new way of smoothing document language models based on probabilistic score propagation in the similarity structure of the corpus which allows us to do the smoothing in *multiple steps*. Our main idea is to propagate word count statistics in a network of similar documents. The network is composed of documents with *generation links* [38] between them. Generation links can be thought of as automatically generated citation links between documents which serve us as alternates for hyperlinks to connect related documents. There is a

generation link between two documents if the language model of the first document gives high probability to the term sequence comprising the second one. The word count statistics are then propagated through the network probabilistically. The intuition behind the propagation is to do word-count propagation between similar documents, smoothing each document by the content of its similar neighbors. The smoothing is performed iteratively, updating the document contents to the point that updating does not affect the content any more. The result will be the smoothed document language models. The iterative nature of the algorithm allows us to smooth each document by the new, *smoothed version* of its neighboring documents, allowing us to propagate term counts to remotely related documents.

We evaluated our algorithm on several TREC data sets, including Associated Press Newswire (AP) 1988, 1989, 1990, the LA Times (LA) and San Jose Mercury News (SJMN). The results show that the proposed algorithm consistently and in most cases significantly outperforms an optimized standard simple collection-based smoothing algorithm (i.e., Dirichlet prior). The results also show that our algorithm is especially effective for improving precision in the top-ranked documents through "filling in" missing query terms in the relevant documents. Compared with other smoothing methods that also exploit local corpus structures, our method is also more effective for improving precision in the top-ranked documents. Since a user often reads only a few top-ranked results in most search engine applications, the proposed smoothing method can be expected to deliver better utility to the users than these existing smoothing methods. Furthermore, our method is shown to be complementary with pseudo feedback which tends to improve the average precision, and a combination of our method and pseudo feedback achieves better performance than either one alone.

The rest of the chapter is organized as follows. We first present some background on language model smoothing and some previous work on smoothing in Section 4.2. We then introduce our probabilistic term propagation algorithm in Section 4.3. We discuss the experiment results in Section 4.4, review the related work in Section 4.5 and finally conclude in Section 4.6.

4.2 Document Language Model Smoothing

4.2.1 Language Modeling Approaches to Information Retrieval

The language modeling approach to information retrieval has been studied extensively in the past few years and has been shown to be successful for many retrieval tasks, such as ad hoc retrieval [60, 49, 33, 102, 41], structured document retrieval [55], distributed information retrieval [80], and expert finding [3, 19]. The basic idea of this approach is to estimate a language model for each document and use the language model to rank the documents given a query.

In the *query likelihood scoring* method [60, 102], the documents are ranked based on the likelihood of the query given each document language model:

$$p(q|d) = \prod_{i=1}^{n} p(q_i|d)$$

where $q = q_1 \dots q_n$ is the query. Thus the retrieval problem is reduced to the estimation of a unigram document language model $p(.|d)$.

In this scoring method, exploiting feedback documents to improve the ranking accuracy is difficult. The *Kullback-Leibler(KL) divergence* scoring method [39] overcomes this problem by introducing the query language model and scoring the documents based on the KL-divergence of the query language model and the document language model:

$$D(q||d) = \sum_{w \in V} p(w|q) log \frac{p(w|q)}{p(w|d)}$$

where V is the set of all words in the vocabulary. Note that the query likelihood method is a special case of the KL-divergence method when the query language model is estimated based on the empirical query word distribution. The estimation of the query model in this method can be improved using feedback models [41, 102].

In both query likelihood and KL-divergence scoring methods, the estimation of the document

language model is an important factor which can affect retrieval performance significantly [102]. In particular, smoothing has been shown to be critical in accurately estimating a document language model. Indeed, when estimating the document language model, the maximum likelihood estimator estimates the probability of each word based on the relative frequency of the word:

$$p_{ML}(w|d) = \frac{c(w,d)}{|d|}$$

where $c(w,d)$ is the number of occurrences of the word w in document d and $|d|$ is the total number of words in d. Thus the maximum likelihood estimator assigns zero probability to those words not occurring in the document which is an underestimation of the probabilities of the missing words. The goal of smoothing is to adjust the maximum likelihood estimate to improve the accuracy of word probability estimation and to avoid the problem of zero probability.

4.2.2 Traditional Smoothing Methods

A general smoothing scheme followed by most traditional smoothing methods involves making the probability of an unseen word proportional to the probability of the word given by a reference language model estimated using the entire collection. The *Jelinek-Mercer(JM)* smoothing method and *Bayesian Smoothing using Dirichlet Priors* are two traditional methods commonly used for smoothing document language models [103].

In the JM smoothing method, the document language model is estimated based on a fixed coefficient linear interpolation of the maximum likelihood model of the document and the global collection model:

$$p(w|d) = (1-\lambda)p_{ML}(w|d) + \lambda p(w|\mathcal{C})$$

where coefficient λ controls the influence of each model.

In the Bayesian smoothing approach (also referred to as Dirichlet prior smoothing), the docu-

ment language model is estimated as:

$$\begin{aligned} p(w|d) &= \frac{c(w,d) + \mu p(w|\mathcal{C})}{|d| + \mu} \\ &= \frac{|d|}{|d| + \mu} p_{ML}(w|d) + \frac{\mu}{|d| + \mu} p(w|\mathcal{C}) \end{aligned}$$

where μ is the Dirichlet prior parameter. This method again involves an interpolation of the individual document model and the collection model, but the coefficient controlling the influence of each model is document-dependent.

One deficiency of these traditional smoothing methods is that the global collection information does not reflect the specific content of individual documents, thus it only provides a crude way for smoothing. To address this deficiency, some recent work [43, 37, 85] has attempted to use the local structure for smoothing with the intuition that the local structure can provide more focused information for better estimation of a document language model. We now briefly review this line of work.

4.2.3 Using Local Corpus Structures for Smoothing

Kurland and Lee in [37] propose to combine the information drawn from the content of the document with how the document is situated within the similarity structure of the corpus to better represent the document. In their method, they use clusters as a means to represent the similarity structure of the corpus. They first construct a set of overlapping clusters of similar documents offline. At retrieval time, they choose a set of appropriate clusters based on the query and smooth the language model of the document with the cluster language model, with the intuition that clusters provide smoothed, representative statistics for their elements. For example, a document belonging to a cluster whose components generally contain the query terms should be considered relevant even if it does not contain the query terms itself. Although in this work, no explicit smoothed document language models are computed, their method essentially achieves the goal of exploiting cluster information to smooth a document language model through their ranking method.

Liu and Croft [43] also smooth representations of individual documents using the corresponding cluster models. They first do either query independent static clustering or query-specific clustering to construct the clusters[1] and build language models for the clusters:

$$p(w|Cluster) = (1-\beta)p_{ML}(w|Cluster) + \beta p(w|\mathcal{C})$$

where β is a general parameter for smoothing and then smooth representations of individual documents using models of the clusters they come from:

$$\begin{aligned} p(w|d) &= (1-\lambda)p_{ML}(w|d) + \lambda p(w|Cluster) \\ &= (1-\lambda)p_{ML}(w|d) + \lambda[(1-\beta)p_{ML}(w|Cluster) + \beta p_{ML}(w|\mathcal{C})] \end{aligned}$$

where λ and β are general parameters for smoothing. In other words, they first smooth the cluster model with the whole collection model and then smooth the document model with the smoothed cluster model. This method is called CBDM for *Cluster-Based Document Model*.

In another study, Tao et. al [85] expand documents using local corpus structures to better estimate document language models. They augment a document probabilistically with potentially all similar documents in the collection. For each document, they construct a probabilistic neighborhood of similar documents where each neighbor is associated with a probability value that reflects how likely it is from the underlying distribution of the original document. They then expand each document with the probabilistic neighborhood around it:

$$c(w,d') = \alpha c(w,d) + (1-\alpha) \sum_{b \in \mathcal{C}-\{d\}} (\gamma_b(d) \times c(w,b))$$

Here d' is the expanded version of d, α is a parameter that controls the balance between the content of the document and the influence of the neighborhood and $\gamma_b(d)$ is the confidence value assigned to each neighboring document b based on its similarity to the document d. They use

[1]The clusters do not overlap in this method.

d', the expanded version of the document, to estimate the document language model. From the smoothing viewpoint, this work is an extension of Liu and Croft's work where each document has its own cluster for smoothing. In the rest of the paper, we will refer to this method as DELM for *Document Expansion Language Model*.

As can be seen, what all these methods do is a one-step interpolation of the document language model and a reference language model. In the following section, we introduce our proposed smoothing method, which propagates scores in the similarity structure of the corpus probabilistically and allows us to do the smoothing in multiple steps.

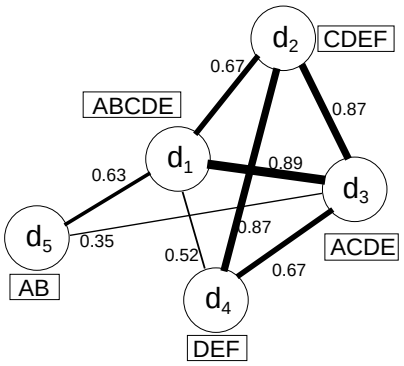
4.3 A Term Propagation Smoothing Method

In this section, we present the term propagation smoothing method. We first discuss why multiple-step smoothing is potentially advantageous over single-step smoothing.

4.3.1 One-Step versus Multiple-Step Smoothing

The current smoothing methods all do one-step smoothing. That is, the smoothed language model of a document is generally a one-step interpolation of the relative frequencies of words in the target document and those in some reference set of documents (either surrounding documents by some similarity or the whole set of documents). Intuitively, we could do such one-step smoothing multiple times. Indeed, if we believe that a smoothed document language model is a better representation of the document than its maximum likelihood estimate (i.e., relative frequencies), then smoothing the language model of a document using the already smoothed language models of its surrounding documents can be better than smoothing using the unsmoothed language models of those surrounding documents. We now use a simple example to illustrate this intuition. Among all the methods which use local information for smoothing, our work is most similar to the DELM method [85]. We thus use this method in the illustration.

Suppose that we have a document collection of five documents, $\mathcal{C} = \{d_1, d_2, d_3, d_4, d_5\}$:



(a) Cosine Similarity Graph

w	*p(w|d5')*
A	0.3
B	0.25
C	0.15
D	0.15
E	0.15
F	0

(b) Smoothed Document LM (DELM Method)

w	*p(w|d5')*
A	0.38085
B	0.32016
C	0.10636
D	0.09628
E	0.09628
F	0.00008

(c) Smoothed Document LM (Propagation Method)

Figure 4.1: One-step versus multiple-step smoothing

DocID	Content
d_1	A B C D E
d_2	C D E F
d_3	A C D E
d_4	D E F
d_5	A B

In order to augment the documents, the DELM method constructs a graph of documents with documents as nodes and cosine similarities as relation weights. Figure 4.1(a) shows the corresponding graph.

It then expands each document by the content of the surrounding documents:

$$c(w, d') = \alpha c(w, d) + (1 - \alpha) \times \sum_{b \in \mathcal{C} - \{d\}} (\gamma_d(b) \times c(w, b))$$

where $c(w, d)$ is the count of word w in document d and $\gamma_b(d)$ is the confidence value assigned to each document b in the neighborhood of d based on the similarity of b and d. $\gamma_d(b)$ controls the influence of b on the expanded version of d. (More details on this method can be found in [85].)

When augmenting d_5 using the DELM method, the only documents influencing d_5 will be d_1

and d_3. The corresponding augmented document language model (assuming α to be 0.5) is shown in Figure 4.1(b). i.e., the probability of the word 'F' in d'_5, the expanded version of d_5, will still be 0.

In multiple-step smoothing, on the other hand, d_5 would be influenced by all d_1, d_2, d_3 and d_4, and the probability of the word 'F' in the smoothed language model for d_5 would be non-zero. Indeed, the smoothed language models for d_1 and d_3 would have a non-zero probability for 'F' after one step of smoothing with d_2. Thus after another iteration of smoothing, in which d_5 would be smoothed with the smoothed languages of its two neighbors d_1 and d_3, the probability of 'F' for d_5 would also be non-zero. That is, the count of 'F' in d_2 can be propagated to d_5 through d_1 and d_3. In Figure 4.1(c), we show some sample smoothing result obtained by applying our proposed method to this toy example. Intuitively, this achieves more accurate smoothing than the result shown in Figure 4.1 (b).

4.3.2 Term Propagation Smoothing

The basic idea of the proposed term propagation smoothing method is precisely to allow counts of terms in a document to "spread" to other documents that are "remotely" related in a weighted manner so that we can achieve multiple-step smoothing of document language models. To implement this idea, we first need to construct a document similarity graph through which the counts can be propagated. Figure 4.2 shows a sketch of the proposed term propagation smoothing method. Having a set of documents, at the first step, we estimate an unsmoothed unigram language model for each document. For each query word, we then compute the probabilities $p^0(d|w)$ using the Bayes' formula. At the third step, we propagate these probabilities in the similarity graph of the documents until they converge. We will show later in this section that with the way we construct the similarity graph and propagate the probabilities will guarantee that the probabilities will converge to a unique probability distribution. Having the new $p^n(d|w)$, we finally estimate the document language model $p_{smooth}(w|d)$ by applying Bayes' rule again. In the following, we present the details of constructing the similarity graph and different steps of the algorithm.

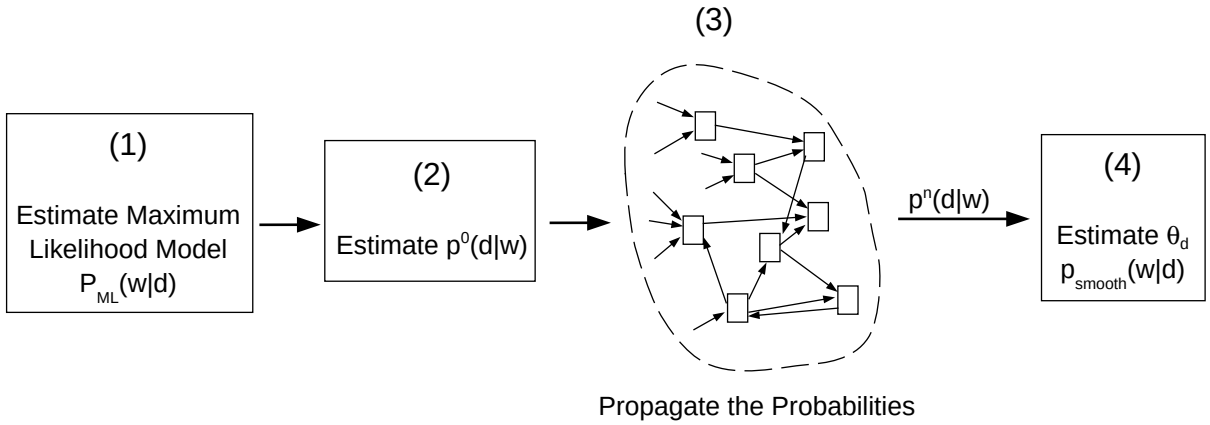


Figure 4.2: Smoothing process steps

Constructing the Generation Graph

The existence of human-created hyperlinks in a hyperlinked environment provides a huge amount of latent judgments about the relevance of documents [36]. However in a non-hypertext setting, these judgments are not available. The problem of automatically generating links between documents in a non-hypertext environment has been studied before [23, 26, 95, 38]. In this work, we use generation graphs proposed in [38] to construct a graph of documents for propagating term counts. A generation graph can be viewed as a graph of documents that cite each other, where the weighted links are induced automatically from the content of the documents. Specifically, a generation graph is a directed graph where documents are the nodes and link weights are proportional to the generation probabilities, the probabilities assigned by the language model of one document to the text of another.

Given any set of documents D, we can construct a generation graph $G = (D, W)$ as follows. For each document $d \in D$, we compute $p(d|g)$, the likelihood of document d given any other document $g \in D$ and take the top k documents that give d the highest likelihoods as k neighbors of d in G. We denote this set of documents by $TopGen(d)$. We have an edge between d and g (i.e., $(d, g) \in W$) if and only if $g \in TopGen(d)$. The probability weight of each edge $p(d \rightarrow g)$ is

simply defined as

$$p(d \rightarrow g) = \frac{p(d|g)}{\sum_{g' \in D} p(d|g')} \tag{4.1}$$

where $p(d|g)$ is assumed to be zero if $g \notin TopGen(d)$. Clearly, $\sum_{g \in D} p(d \rightarrow g) = 1$. Intuitively, this means that we have a conditional probability distribution over all the neighbors of d given d, which can be interpreted as giving the probability of "walking" to a neighbor of d from d. Later we will see that such a probabilistic graph allows us to implement our idea of multiple-step smoothing as a random walk model on this graph.

Given a query, intuitively, improving the language models of the top-ranked documents is most interesting as lowly ranked documents would unlikely be relevant. This suggests that we only need to construct the generation graph for a certain number of top-ranked documents based on their retrieval scores. Such a "working set" approach has an additional advantage of reducing computational overhead and regularizing the propagation to avoid over-smoothing. As will be shown later, smoothing with only a small number of top-ranked documents is more robust and tends to perform better than smoothing with many top-ranked documents.

The generation graph constructed this way captures the similarity structure of the corpus. By propagating scores in the graph, we could allow a document d to iteratively receive support of counts of words from those documents g whose $p(d|g)$ is relatively large. Since $TopGen(d)$ and $p(d|g)$ can be pre-computed, such a generation graph can be constructed efficiently during the run time of a query.

The choice of k here is empirical. In the experiments section, we will show the results for different values of k and analyze the sensitivity to this number.

4.3.3 Probabilistic Term Propagation Algorithm

The probabilistic term propagation (PTP) algorithm involves the following four steps:

Step 1: Having the set of documents, we estimate an unsmoothed unigram language model based on a document d using the maximum likelihood estimate given by the relative counts of the

words:

$$p_{ML}(w|d) = \frac{c(w,d)}{|d|}$$

Here w is any word in our vocabulary V (the vocabulary is composed of all the words that appear in at least one document in D) and $|d| = \sum_{w' \in V} c(w', d)$. Note that using maximum likelihood estimator, we will have zero probabilities for all the words absent in the document.

Step 2: For each query word, we then compute the probabilities $p^0(d|w)$ using the Bayes' formula:

$$p(d|w) \propto p(w|d)p(d)$$

where $p(d)$ is the document prior. The reason why we want to reverse the conditional probability is because $p(d|w)$ defines a distribution over all the documents and this allows us to cast multiple-step smoothing as iteratively revising this distribution based on propagation on the generation graph. It is unclear how we could do the same thing with the original conditional probability $p(w|d)$.

Assuming a uniform document prior, we will have:

$$\begin{aligned} p^0(d|w) &\propto p_{ML}(w|d) \\ &= \frac{p_{ML}(w|d)}{\sum_{d_i \in D} p_{ML}(w|d_i)} = \frac{c(w,d)/|d|}{\sum_{d_i \in D} c(w,d_i)/|d_i|} \end{aligned}$$

Since every word w in our vocabulary must appear in at least one document in D, there is at least one $d \in D$ for which $c(w, d_i) > 0$. At this point, for each query word, we have the estimated conditional probabilities of all the documents given the word, with zero probabilities for those documents not containing the word. i.e., the probabilities of the documents not having the word is underestimated. We will see how propagation on the generation graph can improve this estimate.

Step 3: At this step, we smooth the probability of each document given a word with the probabilities of similar documents, with the intuition that both the content of the current document and the content of similar documents can be useful for estimating the probabilities.

Given a word w, we define the probability of each document as:

$$p(d|w) = \alpha p^0(d|w) + (1-\alpha)\sum_{x \in D} p(x|w)p(x \rightarrow d) \tag{4.2}$$

i.e. a linear combination of its content-based probability and the effect of neighbors in the generation graph. Here $p(x \rightarrow d)$ is the weight of the directed edge from x to d in the generation graph which is defined in Equation 4.1. These probabilities are computed iteratively, updating the probability of each document using the updated probabilities of the neighbors until they converge to a limit. At each step, the score of each document is propagated to its outgoing neighbors in the generation graph in a weighted manner, and the score of each document is updated to a combination of the sum of its incoming (propagated) scores and its own content-based score.

The score definition in Equation 4.2 corresponds to the standing probability distribution of a random walk on the generation graph of the documents. Indeed, the smoothing algorithm can be interpreted as follows: Imagine that a random surfer is surfing the set of documents looking for documents related to the word w. At each step, the surfer would either jump to a related document (by following an edge on the graph) with probability $1-\alpha$ or jump to a random document with probability α. If the surfer decides to jump to a related document (from the current document d) the surfer would land on a document g with probability $p(d \rightarrow g)$; otherwise, the surfer would land on a random document g with probability $p^0(g|w)$. The surfer keeps doing this iteratively, jumping to documents looking for documents related to the word. The final score of each document is equal to the standing probability of the surfer on the document.

In order to compute the scores, we construct a matrix $M = \alpha M_0 + (1-\alpha)M_G$ where $M_0(m,n) = p^0(d_n|w)$ and $M_G(m,n) = p(d_m \rightarrow d_n)$. We then compute the probability scores using matrix multiplication: $\vec{P} = M^T\vec{P}$ where $\vec{P}$ is the vector of the probability values. The probability values are computed iteratively until they converge to a limit. The final scores will be the values of the stationary probability distribution of the Markov chain defined by M. We ensure reachability to each document through smoothing the random jump probability $p^0(d|w)$ slightly

with a uniform distribution over all the documents (similar to the uniform jumping probability in PageRank [56], but we give the otherwise unreachable documents a very tiny probability). Thus by the Ergodicity theorem for Markov chains [30], we know that the Markov chain defined by such a transition matrix M must have a unique stationary probability distribution.

Step 4: Having obtained the propagated conditional probabilities $p^n(d|w)$ (after n iterations), we can "convert" them into the desired conditional probabilities $p(w|d)$ of the document language model by using the Bayes' rule again:

$$p_{smooth}(w|d) \propto p^n(d|w)p(w)$$

where $p(w)$ is the word prior. We estimate the word priors from the counts of the words in the entire collection ($p(w|\mathcal{C})$). Since we have done propagation only for query words, we distinguish two cases for computing these probabilities, one where w is a query word ($w \in Q$) and one where it is not ($w \notin Q$):

$$\begin{aligned} p_{smooth}(w|d) &\propto p^n(d|w)p(w|\mathcal{C}) \\ &= \frac{p(d|w)p(w|\mathcal{C})}{\sum_{w_i} p(d|w_i)p(w_i|\mathcal{C})} \\ &= \begin{cases} \dfrac{p^n(d|w)p(w|\mathcal{C})}{\sum_{w\in Q} p^0(d|w)p(w|\mathcal{C}) + \sum_{w\notin Q} p^n(d|w)p(w|\mathcal{C})} & w \in Q \\ \dfrac{p^0(d|w)p(w|\mathcal{C})}{\sum_{w\in Q} p^0(d|w)p(w|\mathcal{C}) + \sum_{w\notin Q} p^n(d|w)p(w|\mathcal{C})} & w \notin Q \end{cases} \end{aligned}$$

$p_{smooth}(w|d)$ gives the smoothed document language model for document d.

In our probabilistic propagation method, the score propagation is computed once for each query word. As discussed earlier, we do not use the whole graph of documents for propagation, but instead we propagate the counts in the top k documents returned by a basic retrieval method, with the intuition that the documents ranked lower than k are unlikely to be relevant. (Indeed, as will be shown later in the discussion of experiment results, it is actually beneficial to restrict

propagation to only the top-ranked documents.) This node pruning also helps us to speed-up the propagation process. Specifically, given a query, we extract the subgraph corresponding to the top k documents returned by a basic retrieval method from the universal generation graph. The universal generation graph corresponds to the whole set of documents and is constructed once offline. The query subgraphs are generally sparse, since the number of outlinks of each document in the generation graph is prespecified and is commonly small compared to k. Thus we can make use of sparse matrix multiplication methods to speed up the iterative multiplications. Even if we do not exploit sparse matrix multiplication methods, the computational complexity in each iteration of propagation is $O(k^2)$, which is about the same complexity as doing query-specific clustering (with pre-computed similarity matrix) as done in some previous work [43]. In practice, the scores converge to a limit quite fast and the whole propagation process can be done in real time. In our experiments, the propagation took us less than 0.1 seconds (for $k = 1000$) to converge for each query word on a Linux desktop machine with dual Pentium 4 3.0GHz processors and 1GB memory, thus the probabilistic propagation smoothing algorithm is efficient enough to be performed in real-time.

Connection to the General Probabilistic Score Propagation Framework

The propagation step of the probabilistic term propagation method proposed here is obviously a special case of the general probabilistic score propagation framework proposed in Chapter 2. Applying the general framework on the generation graph of documents results a term propagation model with the updating formula:

$$p(d|w) = \alpha \sum_{i=1}^{2} \sum_{x \in D} p(x|w) p_i(x \rightarrow d)$$

$$p_1(x \rightarrow d) = p^0(d|w), \quad p_2(x \rightarrow d) = p(x \rightarrow d)$$

$$\alpha_1 + \alpha_2 = 1$$

Note that in this propagation model, each document has two sets of neighbors: similar document connected through the generation links and the whole set of documents and that the propagated scores are the probabilities of documents given a word. Rewriting this updating formula gives us:

$$\begin{aligned} p(d|w) &= \alpha \sum_{x \in D} p(x|w) p_0(d|w) + (1-\alpha) \sum_{x \in D} p(x|w) p(x \rightarrow d) \\ &= \alpha p^0(d|w) + (1-\alpha) \sum_{x \in D} p(x|w) p(x \rightarrow d) \end{aligned}$$

which is clearly the updating Equation 4.2 in the term propagation method.

4.3.4 Retrieval using the Smoothed Language Model

As discussed in [104], smoothing plays two distinct roles in retrieval. The first role is to improve the accuracy of estimation of document language models. The second is to model any noise in the query. Our propagation method aims at improving smoothing for the first purpose. Thus to ensure that we also model noise in the query, we further perform a second stage of smoothing. That is, we use the following final language for retrieval with the query likelihood retrieval method or the KL-divergence retrieval method:

$$p'(w|d) = \frac{|d|}{|d| + \mu} p_{smooth}(w|d) + \frac{\mu}{|d| + \mu} p(w|\mathcal{C})$$

where μ is a parameter similar to the one in Dirichlet prior smoothing [103]. We set $\mu = 1800$.

Table 4.1: Data sets

Collection	Contents	# of Docs	Queries	Total # of Relevant Docs
AP 89	Associated Press Newswire 1989	84678	1-50	1598
AP 88-89	Associated Press Newswire 1988, 1989	164597	101-150	4805
AP	Associated Press Newswire 1988, 1989, 1990	242918	51-150	21819
LA	the LA Times	131896	301-400	2350
SJMN	San Jose Mercury News	90257	51-150	4881

4.4 Experiments

4.4.1 Data Sets and Baseline Method

As our data sets, we used five TREC test collections: three combination of the Associated Press Newswire 1988, 1989, 1990, the San Jose Mercury News and the LA Times [1] which are the collections previously used for evaluating various smoothing methods [43, 37, 85]. Statistics of the data sets and the queries we used in our experiments are given in Table 4.1. We used the query likelihood method with Dirichlet prior smoothing as our baseline.

4.4.2 Term Count Propagation

Having the baseline ranked list of results, we pick the top "k" documents (50 in our experiments) and extract the similarity graph of this set of documents. The similarity graph could have been constructed in different ways. We use the generation graph with fixed number of neighbors for each document in our experiments. We experimented with different number of neighbors, ranging from 5 to 30.

We then apply our term propagation smoothing method on this set of documents to get the corresponding smoothed document language models. At this step, the parameter α (in the propagation formula (4.2)) allows us to control the amount we want to trust the propagated weights. We changed the value of α from 0.1 to 0.9 in our experiments.

We then put these documents back in the pool of documents and rank the whole data set again and compare this new ranking with the baseline ranking. As the measures of comparison, we report precision at 0.1 recall(Prec@0.1 Recall), precisions at 5 and 10 documents (Prec@5, Prec@10) and Mean Average Precision(MAP).

4.4.3 Basic Results

The first research question we want to answer is whether the proposed term propagation smoothing algorithm would perform better than the baseline Dirichlet prior smoothing method which does not exploit local corpus structure.

In order to answer this question, for each query, we pick the top 50 documents of the query likelihood ranking, extract the corresponding generation graph using 5, 10, 20 and 30 neighbors and do the propagation on these documents. We then rank all the documents in the data set again, using the new smoothed document language model if the document is among the top 50 (other documents are smoothed using the Dirichlet prior smoothing method just as in the baseline). Finally we compare the results with the baseline method. The results are shown in Table 4.2. In the table for each data set, we report the baseline scores as well as the scores of our propagation method with the specified parameters and the amount of improvement we get when using the proposed method. We did a Wilcoxon signed rank test at 0.05 level of significance to see if the improvement is statistically significant. Statistically significant improvements are distinguished by a star (*). We also report the number of relevant retrieved documents and the total number of relevant documents for each experiment (RelRet/TotalRel).

As can be observed from the results, in all the five data sets, we can improve almost all the measures over the baseline, although the improvement shown towards the top of the ranking is more significant than the average improvement shown.

Figure 4.3 shows the Precision-Recall curve for the Baseline as well as term propagation smoothing results for one of our experiments on the SJMN data set. The curve confirms our observation of improvement on top ranks rather than on average where we can see improvement

Table 4.2: Term propagation results versus Dirichlet baseline

		Baseline	PTP	Improvement
AP89	Prec@0.1 Recall	0.384	0.38	-
	Prec@5	0.284	0.316	11.3%
	Prec@10	0.247	0.278	12.6%
	MAP	0.225	0.228	1.3% *
	RelRet/TotalRel	936/1598	937/1598	1 doc.
AP88-89	Prec@0.1 Recall	0.459	0.489	6.5%
	Prec@5	0.412	0.476	15.5% *
	Prec@10	0.384	0.448	16.7% *
	MAP	0.239	0.247	3.3% *
	RelRet/TotalRel	3208/4805	3210/4805	2 docs
AP	Prec@0.1 Recall	0.446	0.463	3.8% *
	Prec@5	0.453	0.533	17.7% *
	Prec@10	0.444	0.497	11.9% *
	MAP	0.223	0.228	2.2% *
	RelRet/TotalRel	10518/21819	10531/21819	13 docs
LA	Prec@0.1 Recall	0.489	0.516	5.5%
	Prec@5	0.347	0.373	7.5%
	Prec@10	0.29	0.31	6.9% *
	MAP	0.247	0.255	3.2%
	RelRet/TotalRel	1625/2350	1627/2350	2 docs
SJMN	Prec@0.1 Recall	0.384	0.442	15.1% *
	Prec@5	0.351	0.406	15.7% *
	Prec@10	0.306	0.367	19.9% *
	MAP	0.204	0.211	3.4% *
	RelRet/TotalRel	3088/4881	3088/4881	-

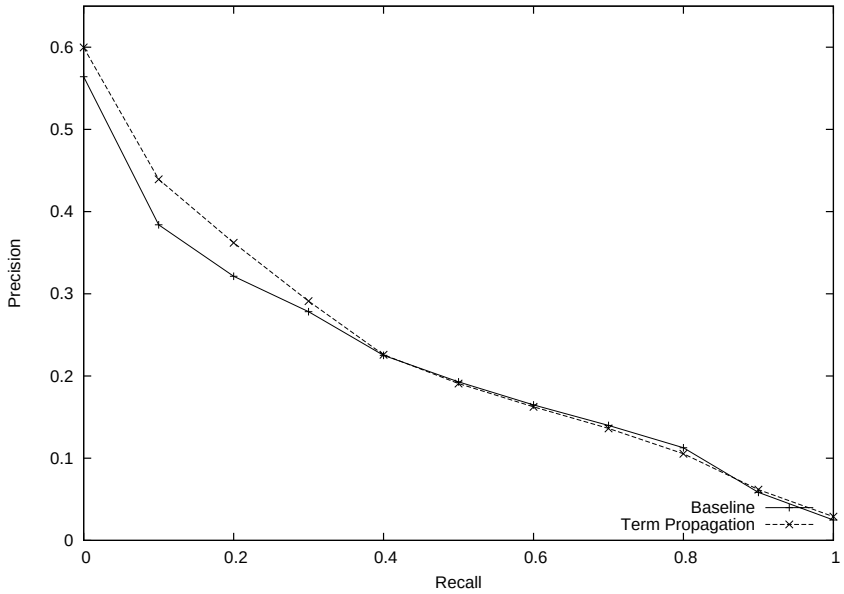


Figure 4.3: Precision-Recall curve for one experiment in SJMN

of our method on the front part of the curve.

This observation is indeed very interesting in that the precision at top ranks is improved even when MAP is not improved that much. This behavior is clearly quite beneficial in any search engine application because a user often only views a small number of top-ranked results.

4.4.4 One-Step versus Multiple-Step Smoothing Results

A major research question we want to answer is whether multiple-step smoothing is more effective than one-step smoothing. We answer this question by looking into the effect of varying the number of iterations in the propagation component of our smoothing algorithm.

We first compare the no-propagation results with the fully-converged results obtained from multiple iterations of propagation in Table 4.3. The no-propagation ranking results is different from the Dirichlet prior baseline because we use the Bayes' rule to compute $p(w|d)$. This comparison helps us to see how much improvement we actually get from propagation. Again we did a Wilcoxon signed rank test at 0.05 level of significance to see if the improvements are significant. Significant improvements are distinguished by a star (*). As the table shows, in all the five data

Table 4.3: Term propagation results versus no-propagation results

		No Propagation	PTP	Improvement
AP89	Prec@0.1 Recall	0.36	0.38	5.6%
	Prec@5	0.267	0.316	18.4%
	Prec@10	0.227	0.278	22.5% *
	MAP	0.217	0.228	5.1% *
AP88-89	Prec@0.1 Recall	0.464	0.489	5.4%
	Prec@5	0.428	0.476	11.2%
	Prec@10	0.38	0.448	17.9% *
	MAP	0.242	0.247	2.1% *
AP	Prec@0.1 Recall	0.449	0.463	3.1% *
	Prec@5	0.452	0.533	17.9% *
	Prec@10	0.438	0.497	13.5% *
	MAP	0.225	0.228	1.3%
LA	Prec@0.1 Recall	0.49	0.516	5.3%
	Prec@5	0.347	0.373	7.5%
	Prec@10	0.292	0.31	6.2% *
	MAP	0.246	0.255	3.7%
SJMN	Prec@0.1 Recall	0.397	0.442	11.3%
	Prec@5	0.362	0.406	12.2% *
	Prec@10	0.309	0.367	18.8% *
	MAP	0.209	0.211	1%

sets, we get significant improvement over the no-propagation method, suggesting that propagation indeed helps improve the accuracy of smoothing.

We further compare the fully-converged results with the results obtained from one-step of propagation in Table 4.4. In one step propagation, we start from the non-smoothed probabilities ($p^0(d|w)$) and do the smoothing with immediate neighbors only, while complete propagation allows us to smooth the documents with remotely related documents. Thus comparing them would allow us to see how much gain we can obtain through involving remotely related documents in smoothing. From the results in Table 4.4, we see that smoothing with remotely related neighbors indeed improves over smoothing with only immediate neighbors in all the data sets except for AP88-89 where the performance of complete propagation is slightly worse than that of one step propagation. We also did a Wilcoxon signed rank test to see if the improvement is statistically significant. Statistically significant improvements are distinguished by '*', '**' and '***' for sig-

Table 4.4: One step propagation versus complete propagation

Data Set	Propagation	Prec@0.1 Recall	Prec@5	Prec@10	MAP
AP89	One Step	0.3691	0.2978	0.2556	0.2175
	Complete	**0.38**	**0.316**	**0.278 ****	**0.228 ****
AP88-89	One Step	0.495	0.48	0.446	0.2483
	Complete	0.489	0.476	**0.448**	0.247
AP	One Step	0.4589	0.5091	0.4939	0.2219
	Complete	**0.463**	**0.533 ****	**0.497**	**0.228**
LA	One Step	0.4768	0.3551	0.2857	0.24
	Complete	**0.516 *****	**0.373 ****	**0.31 *****	**0.255 *****
SJMN	One Step	0.4258	0.3809	0.3415	0.2086
	Complete	**0.442 ****	**0.406 ****	**0.367 ****	**0.211 ***

nificance levels 0.1, 0.05 and 0.01 respectively. In most cases, the improvement is statistically significant. Overall, smoothing with remotely related documents is clearly beneficial.

4.4.5 Comparison with Other Smoothing Methods using Local Corpus Structures

We further compare our method with some other smoothing methods proposed in the previous work that also exploit local corpus structures.

In Table 4.5 we show the PTP results compared with "DELM + Diri" proposed by Tao and others [85] on two of the data sets for which we have complete results of DELM+Diri. As the table shows, in both data sets, we improve precision on top rank results substantially with slightly worse MAP.

In Figure 4.4 we compare our method with CBDM proposed by Liu and Croft [43] on the AP data set based on precision at different recall levels. (We do not have other results of this method.) Again our method slightly outperforms CBDM at low recall values (the front part of the curve) but is slightly worse at high recall levels, confirming that our method tends to improve precision on the top-ranked documents.

Indeed, from Table 4.6, where we compare our method with CBDM based on the mean average

Table 4.5: Comparison with DELM

		DELM+Diri.	PTP	
LA	Prec@0.1 Recall	0.4901	0.5156	(5.2%)
	Prec@5	0.3408	0.3735	(9.6%)
	Prec@10	0.2867	0.3112	(8.2%)
	MAP	0.2655	0.2547	(-)
SJMN	Prec@0.1 Recall	0.4023	0.4559	(13.3%)
	Prec@5	0.3617	0.4170	(15.3%)
	Prec@10	0.3245	0.367	(13.1%)
	MAP	0.2266	0.2201	(-)

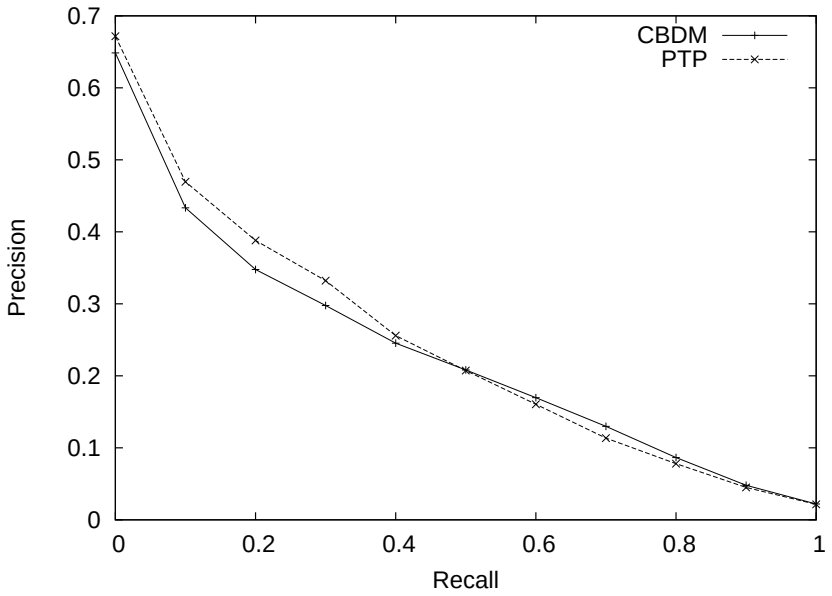


Figure 4.4: Comparison with CBDM

Table 4.6: Comparison with CBDM based on MAP

	CBDM	PTP
AP	0.2326	0.23
LA	0.259	0.2547
SJMN	0.2171	0.2201

precision, we see that our MAP values are comparable to the CBDM results.

It is quite interesting to see that in all these results, our method outperforms these other methods in precision of the top-ranked documents, but does not really improve the MAP; indeed, the MAP is often slightly worse. This observation motivates us to look into the reason why our method appears to be especially good at improving precision of the top-ranked documents, and we find that it is likely because our method can help those relevant documents missing at least one query term to "fill in" the missing query terms through iterative propagation, thus improving their ranks. Indeed, the multiple-step smoothing mechanism of our proposed algorithm allows term counts to be propagated to those remotely related documents. We now present a more detailed analysis of term propagation smoothing in this line and examine its sensitivity to various parameters.

4.4.6 Detailed Analysis of Term Propagation Smoothing Algorithm

Understanding the Improvement in Precision of Top-Ranked Documents

Since a main motivation of term propagation smoothing method is to achieve multiple-step smoothing and allow term counts in a document to help smooth those remotely related documents, we hypothesize that the reason why our method appears to be very good at improving precision of top-ranked documents is that our method promotes those relevant documents that do not match all query terms by "filling in" the missing query terms through iterative propagation of term counts. In order to test the hypothesis, we compare our ranking in top 10 with the Dirichlet prior smoothing baseline ranking and take out the unique relevant documents in each ranking. We then count the number of documents missing at least one query term in each set. Table 4.7 shows the results of this comparison for three of the data sets.

Table 4.7: Percentage of relevant documents in top 10 with at least one query word missing

	Unique to Baseline	Unique to PTP
AP88-89	38.2%	48%
LA	19.5%	33.8%
SJMN	24.1%	44.2%

As the table shows, in all the three collections, the percentage of documents with at least one query word missing in our method is much higher than the baseline, suggesting that our hypothesis is true and our method helps the documents with missing query words to come to the top by filling in their missing query word(s). Indeed, according to the clustering hypothesis [92], which states that relevant documents tend to be more similar to each other than to non-relevant documents, our generation graph likely will connect many relevant documents to each other. Thus with our propagation algorithm, we can effectively "borrow" terms from one relevant document to help other relevant documents to fill in the missing query terms even when the document supplying a term is only remotely related to the documents receiving the term. While such propagation may potentially also help non-relevant documents to gain extra counts for query terms, the clustering hypothesis suggests that relevant documents will generally get more help than non-relevant documents through such propagation since most of the counts of query terms are in those highly relevant documents and non-relevant documents are generally not as close to such highly relevant documents as relevant documents are. Thus although we do not perform clustering explicitly, our smoothing method can be regarded as one way to exploit the clustering hypothesis to improve the estimation of language models. The fact that our method can effectively improve precision of top-ranked documents suggests that the clustering hypothesis indeed holds for the top-ranked documents. However, a detailed analysis of the performance of our method suggests that the clustering hypothesis may not hold for documents lowly-ranked in the search results (see Section 4.4.6). That is, in the biased sample of lowly-ranked documents, relevant documents are not necessarily more similar to each other than to non-relevant documents.

However, since we propagate $p(d|w)$, when we fill in the missing terms in one document (i.e., one document gets a larger $p(d|w)$), we would inevitably reduce the probability of these terms in

their original documents to maintain the constraint $\sum_d p(d|w) = 1$. This means that the benefit of filling in missing query terms in top-ranked documents may be at the price of pushing down some other relevant documents that are not well-connected with most relevant documents (thus not getting benefit from propagation). This may be the reason why our method is not effective for improving MAP which measures the overall ranking accuracy and especially emphasizes the precisions at high recall levels. Further analysis of the behavior of the propagation algorithm would be a very interesting future research direction.

We now study the sensitivity of the term propagation to some parameters.

Number of Top Documents for Smoothing

In our method, we pick the top "k" documents returned by a basic retrieval method to construct the generation graph for smoothing document language models. We have so far reported the results for smoothing the top 50 documents ($k = 50$). Here we compare the results of smoothing the top 50 documents with the case where we do smoothing on a much larger set of documents, i.e. the top 1000 documents ($k = 1000$). Table 5.3 compares precision at 0.1 Recall, precision at 5 documents, precision at 10 documents and mean average precision for these two cases.

As can be seen from the table, in most cases, both smoothing the top 50 documents and the top 1000 documents outperform the baseline results and smoothing the top 50 documents outperforms smoothing the top 1000 documents. The reason can be that the top 50 documents form a more coherent cluster of documents related to the query compared to the top 1000 documents which may contain many non-relevant documents and the top neighbors of a document may actually not be very similar to the document. Thus propagating using a large graph may not be as reliable as using a small graph and can potentially introduce unreliable propagation.

From the viewpoint of clustering hypothesis, this suggests that relevant documents are more clustered together in the top-ranked documents than in lowly-ranked documents. That is, in the top-ranked documents, relevant documents are very close to each other (making propagation quite effective and reliable), but the relevant documents ranked down in the result list are not necessarily

Table 4.8: Smoothing different number of top documents

		Prec@0.1 Recall	Prec@5	Prec@10	MAP
AP89	Baseline	0.384	0.284	0.247	0.225
	k = 50	0.38	0.315	0.278	0.228
	k = 1000	0.412	0.316	0.28	0.243
AP88-89	Baseline	0.459	0.412	0.384	0.239
	k = 50	0.489	0.476	0.448	0.247
	k = 1000	0.462	0.432	0.412	0.252
AP	Baseline	0.446	0.453	0.444	0.223
	k = 50	0.463	0.533	0.497	0.228
	k = 1000	0.453	0.483	0.464	0.22
LA	Baseline	0.489	0.347	0.29	0.247
	k = 50	0.516	0.373	0.31	0.255
	k = 1000	0.487	0.363	0.302	0.252
SJMN	Baseline	0.384	0.351	0.306	0.204
	k = 50	0.442	0.406	0.367	0.211
	k = 1000	0.408	0.404	0.343	0.21

more similar to each other or to those highly relevant documents, where most of the counts of query terms are, than some highly ranked non-relevant documents are. This observation is consistent with what is observed in some other work exploiting clustering hypothesis. For example, in the work [87], it is found that query-dependent clustering is more effective than static query-independent clustering. Similarly, query expansion with local context analysis (i.e., pseudo feedback) is more effective than with global co-occurrence analysis [98] (pseudo feedback can also be regarded as a way to leverage clustering hypothesis). All this work and our work seem to suggest that the clustering behavior of relevant documents may be more salient in the top-ranked documents than in the entire collection, which intuitively also makes sense as within a biased sample of top-ranked documents relevant documents may form a much more coherent cluster than they do in the entire collection.

Our analysis above also suggests that we should apply the proposed propagation algorithm to a relatively small number of top-ranked documents in real applications, which is actually beneficial in terms of reducing the computational cost.

Parameter α

The parameter α controls the amount of influence from the neighbors in propagation. In Figures 4.5 and 4.6 we show the sensitivity of precision at 5 documents, precision at 10 documents and mean average precision to α for the SJMN and AP88-89 data sets respectively.

As the figures show, the optimal range for good performance towards the top of the ranking is quite wide, showing that our method for term weight propagation is useful with quite a wide range of parameters. The best results are achieved somewhere in the middle. However, a small value of α can really hurt MAP, especially when the number of neighbors is small. This is expected because a small α means mostly relying on the counts from very few neighbors to estimate a language model, likely resulting in quite biased smoothing.

Number of Neighbors

Given a certain number of top-ranked documents to use for constructing a generation graph, we may generate the graph with different numbers of neighbors for each document. Figure 4.7 shows the graphs of precision at 5 documents, precision at 10 documents and mean average precision when propagating through different number of neighbors for each document in the SJMN data set. This parameter is set when we construct the similarity graph and determines the number of documents to which each document propagates its weight. As the figures for precision at 5 and precision at 10 documents show, from some point on, we gradually lose the amount of benefit from word propagation as we increase the number of neighbors. The reason can be that each document has to propagate some of its weight to its neighbors, in the case of large number of neighbors, to potentially non-relevant ones. Thus top relevant documents may be discounted this way, decreasing precision at 5 and precision at 10 documents. On the other hand, propagating to more neighbors will allow low score, hard to reach relevant documents to get some benefit from other documents and move up. That is why the MAP figure shows some improvement when we increase the number of neighbors.

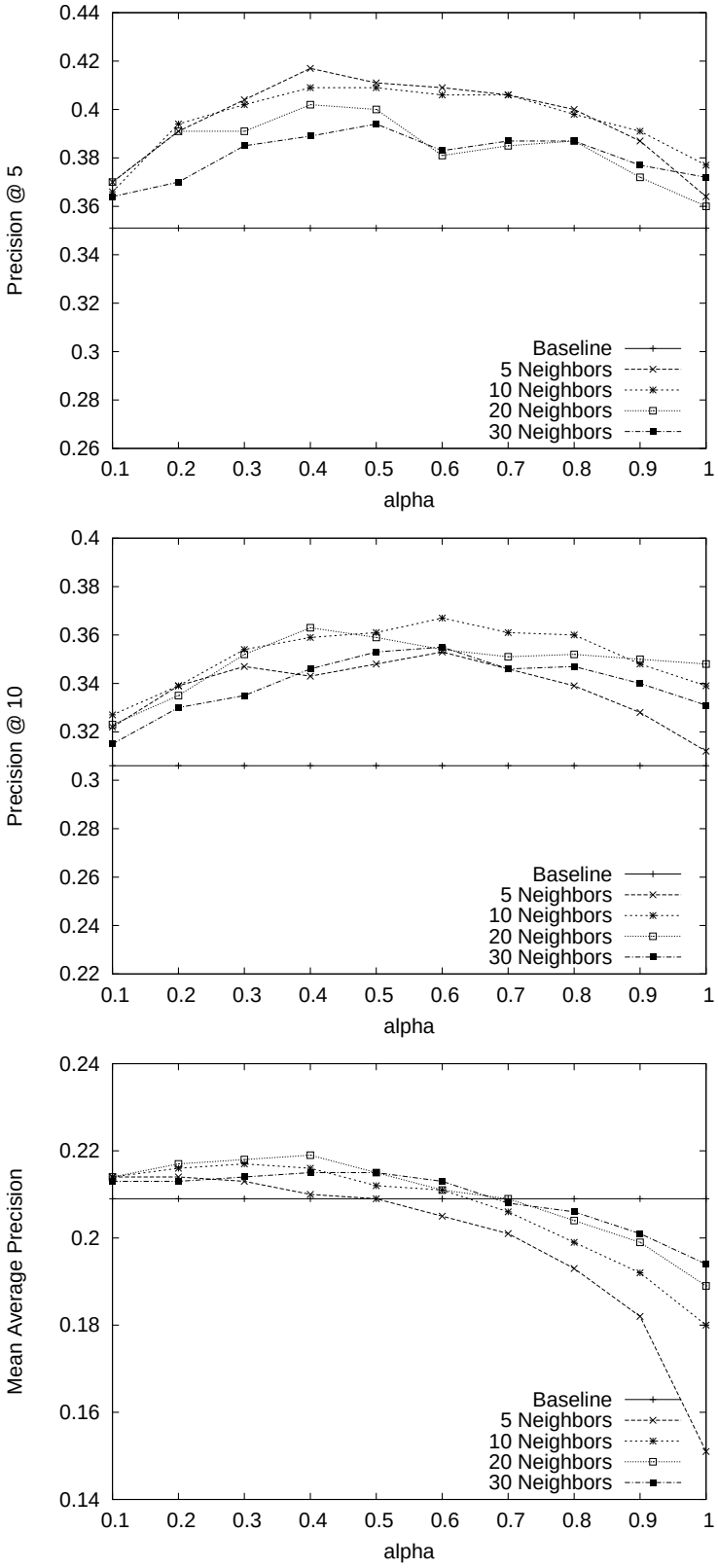


Figure 4.5: Sensitivity to α (SJMN data set)

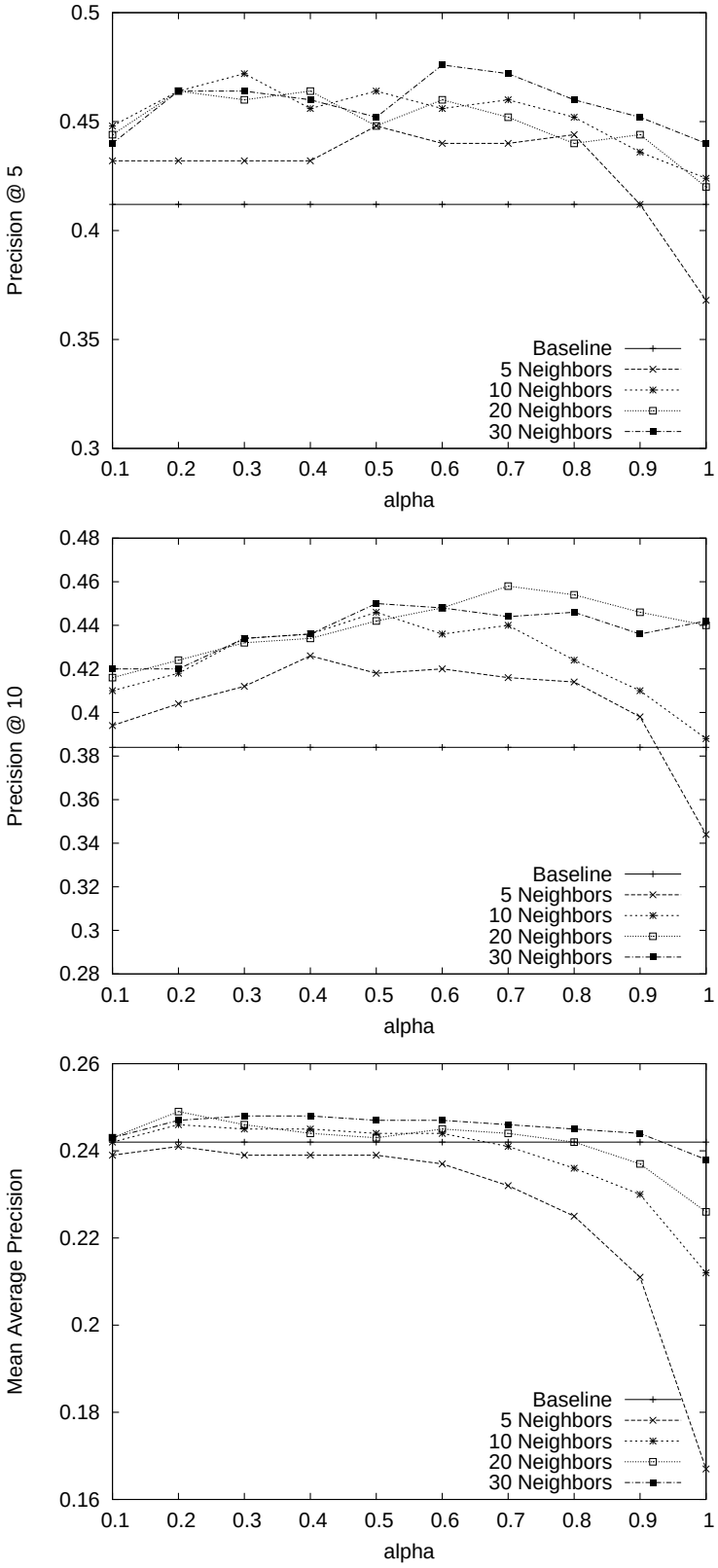


Figure 4.6: Sensitivity to α (AP88-89 data set)

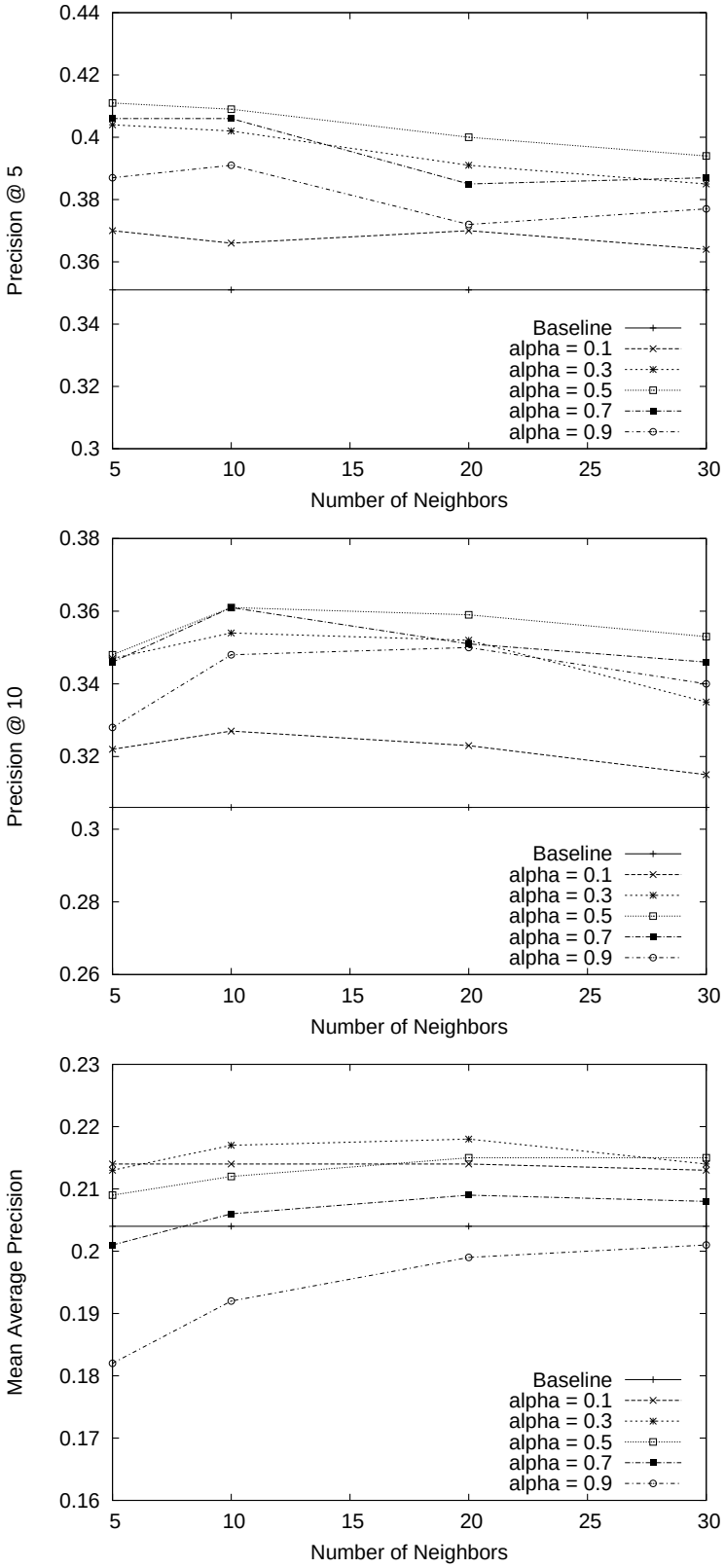


Figure 4.7: Sensitivity to the number of neighbors (SJMN data set)

Table 4.9: Query expansion on top of probabilistic term propagation smoothing

		Baseline	Query Expansion	PTP Smoothing	Query Expansion on top of PTP Smoothing
AP89	Prec@0.1 Recall	0.384	0.403	0.38	0.402
	Prec@5	0.284	0.289	0.316	0.316
	Prec@10	0.247	0.256	0.278	0.278
	MAP	0.225	0.247	0.228	0.26
	RelRet/TotalRel	936/1598	1052/1598	937/1598	1031/1598
SJMN	Prec@0.1 Recall	0.384	0.439	0.442	0.464
	Prec@5	0.351	0.37	0.406	0.413
	Prec@10	0.306	0.341	0.367	0.377
	MAP	0.204	0.245	0.211	0.25
	RelRet/TotalRel	3088/4881	3408/4881	3088/4881	3452/4881

4.4.7 Combination with Query Expansion

Finally we study whether we can further improve retrieval accuracy by combining our smoothing method with query expansion and pseudo feedback. Query expansion has been shown to be an effective way of improving query representation [67, 98, 102]. In our propagation method, we use different information than pseudo-feedback, thus intuitively we should be able to combine these two methods to further improve the performance.

To test this hypothesis, we perform query expansion on top of term propagation smoothing. Specifically we use the top 10 documents after term propagation to perform feedback using the mixture model approach implemented in the Lemur toolkit [102]. The basic idea of this approach is to fit a mixture model to the feedback documents and estimate a feedback topic language model, which is then interpolated with the original query model to generate an "expanded" query model for scoring documents. We used the default settings of all the parameters (i.e., 0.5 for both background noise and feedback coefficient and 20 terms for expanding the query model). Since our method helps to improve the precision at top ranks, we expect to get benefit from this new (improved) ranking for pseudo feedback. Experiment results show that this is indeed true. In Table 4.9, we compare the performance of PTP smoothing, query expansion and their combination for two of our data sets.

The results of the combination show a very interesting feature of the combined algorithm: pseudo feedback usually improves MAP, but the improvement in precision of top-ranked documents is not as much. On the other hand, our method helps more on improving the precision at the top ranks. The combined algorithm has the good features of both, improving both the precision at top ranks and the mean average precision.

4.5 Related Work

Smoothing of document language models has been studied extensively. Most work in this area uses a global background model for the purpose of smoothing [60, 49, 33, 103]. More recent work uses some local corpus structures [43, 37, 85] with the intuition that the local structure can provide more focused information for better estimation of language models. Our work extends all this work in that it considers multiple steps of smoothing and allows smoothing with remotely related documents. As shown in our experiment results, such extension is beneficial.

Our work is related to the clustering hypothesis [92]. The hypothesis states that relevant documents tend to be more similar to each other than to non-relevant documents, and therefore tend to appear in the same clusters. Although we do not perform clustering explicitly, our smoothing method can be regarded as one novel way to exploit the clustering hypothesis to improve the estimation of language models. In this sense, our work is related to some previous work on document clustering [94, 96, 87] and pseudo relevance feedback [98]. It is interesting that in both our study and the work [87], exploiting the corpus structure in documents highly similar to the query is more effective than using a larger working set of documents or the entire collection. This may suggest that the clustering behavior of relevant documents (relative to that of non-relevant documents) may be more salient in the top-ranked documents.

The problem of automatically generating links between documents in a non-hypertext environment has been studied before [23, 26, 95, 38]. We used generation graphs proposed in [38] where the graphs are used to propagate document scores; our work differs from it in that we use the graph

to propagate term counts for smoothing a probabilistic language model.

The idea of using random walks for ranking purposes has also been studied before. For example, PageRank [56] and Topic Specific PageRank [32] are stationary probability distributions for the Markov chain induced by random walks on the Web graph. SALSA [42] examines random walks on the graph derived from the link structure to find authoritative sites on a topic. A recent work [14] has studied random walks on click graphs to produce a probabilistic ranking of documents for a given query.

4.6 Summary

In this chapter, we cast the problem of smoothing document language models as a problem of propagating term counts among documents probabilistically, and presented a novel method for smoothing document language models based on this idea. A major advantage of this method over previous methods is that it provides a principled way to bring in *remotely* related documents to smooth the current document. Evaluation results on several TREC data sets show that the proposed method significantly outperforms the simple collection-based smoothing method and smoothing with remote neighbors in the document similarity graph outperforms smoothing with only immediate neighbors. Compared with other smoothing methods that also exploit local corpus structures, our method is especially effective in improving precision in top-ranked documents through "filling in" missing query terms in relevant documents, which is presumably most important in practical applications as a user often only reads a few top-ranked documents. Furthermore, our method is shown to be complementary with pseudo feedback which tends to improve the average precision, and a combination of our method and pseudo feedback achieves better performance than either one alone.

Although our method consistently improves precision among top-ranked documents, it does not improve the average precision so consistently. A major future research direction is to further study how to improve both the average precision and the precision in top-ranked documents.

Chapter 5

Probabilistic Score Propagation for Cross-Language Information Retrieval

In this chapter, we focus on applying the proposed general probabilistic score propagation framework to do cross-language information retrieval. Cross-language information retrieval has so far been studied with the assumption that some high quality resources such as bilingual dictionaries or parallel corpora are available. Unfortunately since creation of such high quality resources is labor-intensive, they are not always available, especially for minority language pairs. However, resources such as comparable corpora are often naturally available (e.g., news articles published in different languages at the same time period). In this chapter we investigate whether we can perform cross-language information retrieval when the only resource we have is comparable corpora for the language pair. We will apply the general probabilistic score propagation framework to a graph of terms with implicit mutual information links and term correlation links to generate a probabilistic score propagation model for cross-language information retrieval. With the generated probabilistic score propagation model, we iteratively propagate term statistics in the graph of terms to construct the query language model in the target language corresponding to the given query (in the source language). We then retrieve the documents in the target language using the generated query language models. We will show that the proposed method is effective for this task, demonstrating that it is feasible to perform cross-language information retrieval with just comparable corpora.

5.1 Introduction

Cross-language information retrieval (CLIR) is an important technique that can enable universal access to information in all different languages by people speaking different languages. Due to

its importance, cross-language IR has been extensively studied. However, most existing work on CLIR has assumed the availability of at least some reasonable linguistic resources such as a bilingual dictionary or parallel corpora. While such resources may be available for popular languages, they are not available for many pairs of minority languages.

In this chapter, we study how to do CLIR when we have only comparable corpora for the language pair. Comparable corpora are text documents in two different languages that cover similar topics. For example, news articles published in two different languages in the same time period naturally form comparable corpora. Although we may not have reliable linguistic resources such as a bilingual dictionary or parallel corpora for a language pair, we often have comparable corpora, thus assuming the availability of comparable corpora is a realistic assumption.

In this study, we evaluate the feasibility of leveraging some recent work on learning word associations from comparable corpora based on time correlations to do cross-language information retrieval. One challenge here is how to incorporate word correlations into a CLIR model. We study this issue in the language modeling framework. As a basic method, we obtain word translation probabilities based on the time correlations between word pairs and estimate a query language model for the target language, and then use a standard retrieval method to score documents in that language. We study how to effectively transform a time correlation into a probability. We further propose a propagation framework which exploits word co-occurrences in monolingual data as well as time correlations to better estimate the query language models in the target language.

We use the data set used in TREC-2002 [54] Arabic-English retrieval task for evaluation. Our evaluation results show that compared to the monolingual baseline and using the basic CLIR method, we can achieve up to 64.3% of Mean Average Precision(MAP), 67.7% of precision at 5 documents (Prec@5) and 69.4% of precision at 10 documents (Prec@10). Appropriate transformation of raw correlation scores help us to improve the performance to 70.8% of MAP, 70.6% of Prec@5 and 75.3% of Prec@10. The results further show that the proposed probabilistic model is an effective method which helps us to achieve up to 75.9% of MAP, 76.5% of Prec@5 and 77.2% of Prec@10.

The rest of the chapter is organized as follows. We first present some previous work in Section 5.2. Then we introduce our proposed cross-language information retrieval methods in Section 5.3, discuss the experiment results in Section 5.4 and conclude in Section 5.5.

5.2 Previous Work

Cross-language information retrieval deals with finding information in one language in response to a query in another language. Since the query and the documents are expressed in different languages, direct matching of the query and the documents is impossible. Thus some kind of translation should occur before matching is performed. One specific issue in CLIR is where to obtain the translation knowledge [53]. The most common translation resources are bilingual dictionaries, parallel corpora, machine translation systems and comparable corpora. Machine translation systems, bilingual dictionaries and parallel corpora are expensive resources which are not available for many minority language pairs. However comparable corpora are much easier to obtain. Zanettin [99] introduced several available bilingual comparable corpora such as newspaper articles (by section, topic or date), medical articles from journals and textbooks, and tourist brochures and guides. On the other hand, extracting knowledge from comparable corpora is more challenging.

Using comparable corpora as a language resource for cross-language information retrieval has been studied extensively in the existing literature [63, 58, 79, 21, 25, 45, 68, 51, 86]. But most of these works assume some other kind of linguistic resource(s) to be available as well. Picchi and Peters [58], Franz et. al, [21], Fung [25] and Sadat [68] use some kind of bilingual dictionary or bilingual lexical database on top of comparable corpora, Maxuichi et. al, [45] use a small parallel corpus as the training data and Munteanu et. al, [51] require both a bilingual dictionary and a small amount of parallel data. Tao et. al, [86] are among a few which do not use any linguistic resources but comparable corpora. In their work, they exploit frequency correlations of words in different languages in the comparable corpora and discover mappings between words in different languages. In this work, we study how to effectively transform these word mappings into probabilities to do

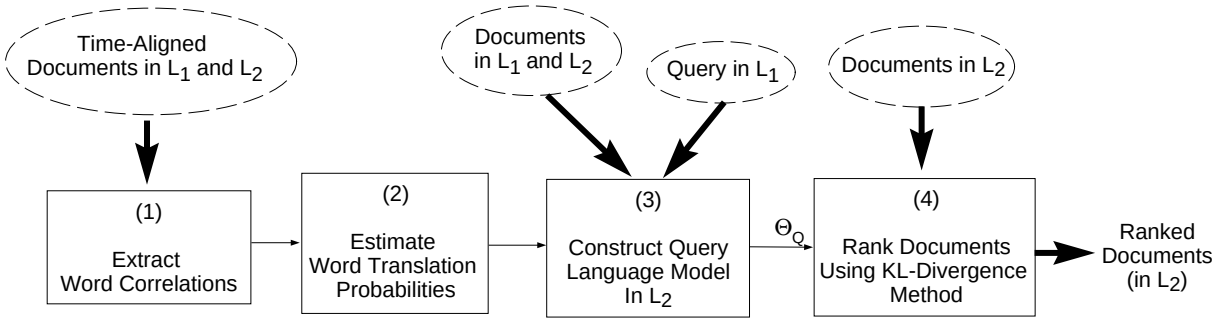


Figure 5.1: Proposed CLIR steps

cross-language IR.

5.3 Cross-Language Information Retrieval with Comparable Corpora

In this section, we present our proposed methods of using learned cross-lingual word associations from comparable corpora to do cross-language information retrieval. As a basic method, we use word correlations mined from comparable corpora to obtain word translation probabilities between word pairs and add them to a standard CLIR method. We study how to effectively transform a time correlation into a probability. We further propose to use a propagation method to exploit word co-occurrences in the monolingual data as well to improve the CLIR performance. Figure 5.1 shows a sketch of our proposed method. In the rest of this section, we will present different steps in more detail.

5.3.1 Extracting Word Correlations

Having time-aligned comparable corpora, we use the method proposed by Tao et.al, [86] to discover correlations between words in different languages. In this method, frequency correlations of words in different languages in the comparable corpora is used to discover mappings between words. The main idea is based on the observation that *the words that are translations of each other*

or are about the same topic tend to co-occur in the comparable corpora at the same time period. Such correlations are exploited to discover the associations of words in different languages.

In this method, each word is represented by a vector of frequencies and each pair of words in different languages is scored based on the similarity of their frequency vectors. The Pearson's correlation coefficient is used to score every word in one language against every word in the other language.

Formally, let $C = \{(d_1, d'_1), \ldots (d_n, d'_n)\}$ be the comparable corpora where d_i and d'_i are documents with the same time stamp in languages L_1 and L_2 respectively. Also let a be a word in L_1 and b be a word in L_2. The normalized frequency vectors for a and b will be $\overrightarrow{a} = (a^1, \ldots, a^n)$ and $\overrightarrow{b} = (b^1, \ldots, b^n)$ respectively where

$$a^i = \frac{c(a, d_i)}{\sum_{j=1}^{n} c(a, d_j)}, b^i = \frac{c(b, d'_i)}{\sum_{j=1}^{n} c(b, d'_j)}$$

and $c(a, d_i)$ is the count of word a in d_i. The similarity of these two words is computed using the Pearson's correlation coefficient:

$$r(a, b) = \frac{\sum_{i=1}^{n} a^i b^i - \frac{1}{n} \sum_{i=1}^{n} a^i \sum_{i=1}^{n} b^i}{\sqrt{(\sum_{i=1}^{n} a^{i2} - \frac{1}{n}(\sum_{i=1}^{n} a^i)^2)(\sum_{i=1}^{n} b^{i2} - \frac{1}{n}(\sum_{i=1}^{n} b^i)^2)}}$$

Figure 5.2 shows a sample set of top English-Arabic word pairs extracted from the English-Arabic comparable corpora we used in our experiments. It is obvious that in most cases, the matching has a very high quality. Note that we had done stemming on the English and Arabic data and that's why some prefixes and/or suffixes are stripped off.

5.3.2 Estimating Word Translation Probabilities

Having the correlations between words in different languages, in the next step we estimate translation probabilities of these words. A natural baseline method is to use the normalized correlation scores as translation probabilities. Formally, let w be a word in L_1 and $u_1, \ldots, u_m$ be the top m

English Word	Arabic Word	Correlation Score
assassin	اغتيال	0.912029
develop	تنم	0.922141
earthquak	زلزال	0.926612
elect	انتخاب	0.924854
gaza	غز	0.907011
isra	اسراييل	0.907502
laden	لادن	0.908982
lebanon	لبن	0.903235
netanyahu	نتانياهو	0.945646
nile	نيل	0.908473
palestinian	فلسطين	0.939951
russia	روسيا	0.914124
secur	امن	0.939588
talk	محادث	0.919655
trade	تجار	0.935074
turkei	تركيا	0.915668
war	حرب	0.960423
world	عالم	0.921579
year	عام	0.920337

Figure 5.2: Sample English-Arabic extracted word pairs

correlated words in L_2 with correlation scores $r_1, ..., r_m$ respectively, i.e., $r_i = r(w, u_i)$, $1 < i < m$ are the top m scores among $r(w, u)$, $u \in V$. We construct the probabilities by normalizing these raw correlation scores:

$$p(u_i|w) = \frac{r_i}{\sum_{j=1}^{m} r_j}$$

where $p(u_i|w)$ is the probability of u_i being the translation of word w in L_2. One deficiency of this naive method is that we trust low correlations too much. Intuitively, high correlations are trustable, but not low correlations. Thus the probabilities should drop sharply as the correlations become smaller.

To take this intuition into account, we decided to transform the scores with a transformation function that penalizes low correlation scores. For the transformation function, we chose to use exponential transformation with the general form:

$$f(x) = ae^{bx} + c$$

We set two restrictions on the transformation function: transform the highest correlation possible (1) to 1 and transform the lowest correlation possible (0) to 0, i.e, $f(1) = 1$ and $f(0) = 0$. Thus we came up with this exponential transformation function:

$$f(r) = \frac{1}{e^b - 1} e^{br} - \frac{1}{e^b - 1}$$

where b is a parameter that controls how much we want to penalize low correlations. Figure 5.3 shows the effect of exponential transformation for different values of b. As can be seen from this figure, higher values of b penalize low correlation values more.

We then construct the probabilities from these converted scores:

$$p(u_i|w) = \frac{f(r_i)}{\sum_{j=1}^{N} f(r_j)} = \frac{\frac{1}{e^b-1} e^{br_i} - \frac{1}{e^b-1}}{\sum_{j=1}^{N} (\frac{1}{e^b-1} e^{br_j} - \frac{1}{e^b-1})}$$

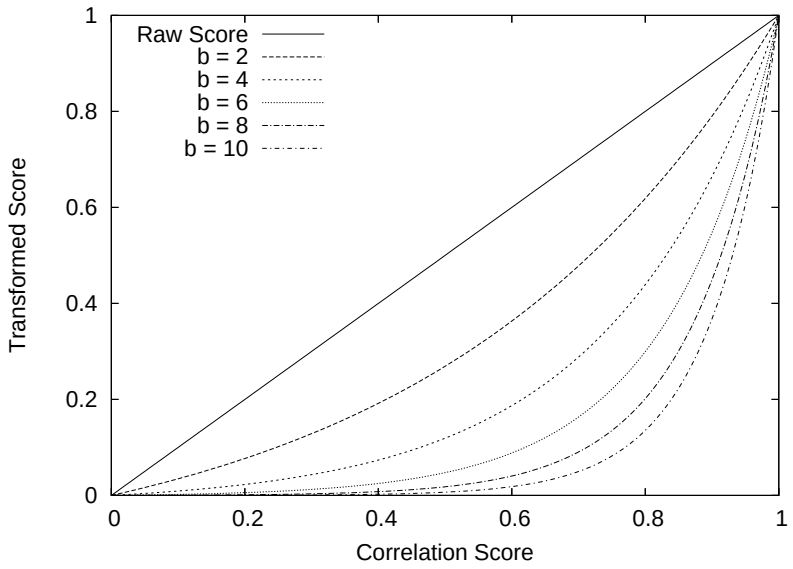


Figure 5.3: Exponential transformation with different values of b

These probabilities drop sharply as the correlations become smaller and thus unreliable.

5.3.3 Constructing Query Language Models and Ranking the Documents

Given the query Q in language L_1, our goal is to find related documents in language L_2. Intuitively, if we can somehow map the query Q in L_1 to a corresponding query in L_2, then we can easily find related documents in L_2 by comparing them to this translated query using a typical retrieval method. Thus what we have to do is to estimate the query language model of the translated query in L_2. Here we propose two methods for constructing the query language model of the translated query. As a basic method, we propose to use the translation probabilities estimated in step 2 directly to translate query words in L_1 to the corresponding words in L_2 and to construct the query language model in L_2 from these translated query words. In our second method, we propose a propagation framework which exploits word co-occurrences in the monolingual data as well to construct the query language model. We will present each method in more detail in the following.

Basic Query Translation Method

Having estimated the translation probabilities between words in the two languages L_1 and L_2, we construct a query language model in L_2 corresponding to the given query Q (in L_1) using our "Top-K translation" method. In this method, for each query word in L_1, we use the top k correlated words in L_2 as its translation and construct the translation of the whole query. We assume all query words to be equally important in this method and thus have equal weights in constructing the query language model. The influence of each translation word depends on the estimated translation probability of the word.

Formally, let $Q = q_1 \dots q_n$. We estimate the query language model in L_2 using:

$$p(w|\hat{\Theta}_Q) = \sum_{i=1}^{n} \frac{1}{n} \frac{p(w|q_i)}{\sum_{j=1}^{k} p(w_j|q_i)}$$

where $p(w_l|q_j) > 0$ if w_l is in the top k correlated words of q_j and $p(w_l|q_j) = 0$ otherwise. This way we have constructed a naive translation of the query in L_2.

Propagation Method

In the basic proposed method, as the translated query words, we only consider those words in L_2 that are highly correlated with the query words in L_1 and use the translation probabilities to construct the query language model. But we observe that we can also consider word co-occurrences in the monolingual data to better estimate the query language models. Using co-occurrence information can introduce related words to the queries in both languages, resulting a better estimation of query language models. Intuitively, a word has a high chance of being in the translated query if it is highly correlated with a word in the source language which has a high probability of being in the query language model or/and it co-occurs a lot with a word in the same language that has a high probability of being in the translated query.

To implement this idea, we first need to construct a network of related words. We construct a network of all the words in language L_1 and all the words in language L_2 where the edges

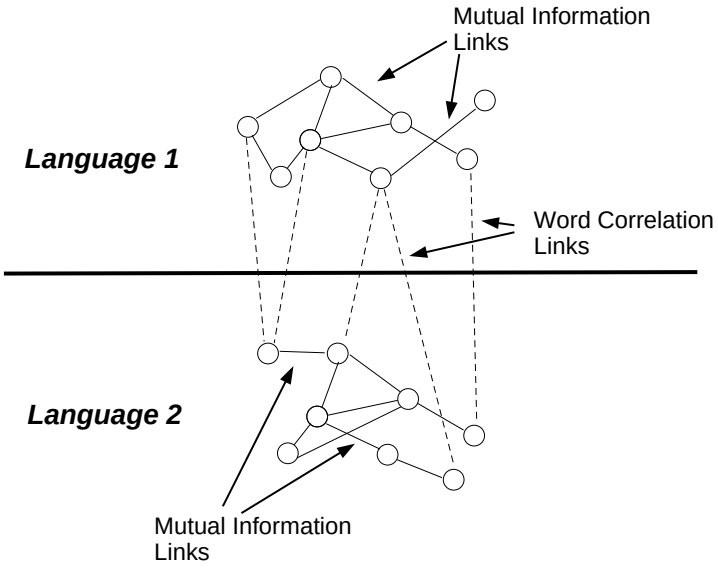


Figure 5.4: Word network structure

between words in the same language show the mutual information between words and the edges between words in different languages are correlation edges. The structure of the network is shown in Figure 5.4. A word has a high score in this network if it is surrounded by words with high scores.

In this network, we define the probability of each word as:

$$\begin{aligned} p(w^1) = \alpha_0 \quad & p_0(w^1) \\ & + \alpha_{MI} \sum_{x \in L_1} p(x) p_{MI}(x \rightarrow w^1) \\ & + \alpha_{trans} \sum_{y \in L_2} p(y) p_{trans}(y \rightarrow w^1) \qquad \text{if } w^1 \in L_1 \end{aligned} \tag{5.1}$$

$$\begin{aligned} p(w^2) = \alpha_0 \quad & p_0(w^2) \\ & + \alpha_{trans} \sum_{x \in L_1} p(x) p_{trans}(x \rightarrow w^2) \\ & + \alpha_{MI} \sum_{y \in L_2} p(y) p_{MI}(y \rightarrow w^2) \qquad \text{if } w^2 \in L_2 \end{aligned} \tag{5.2}$$

$$\alpha_0 + \alpha_{MI} + \alpha_{trans} = 1$$

i.e. a combination of its translation probability and the effect of neighbors in the word network. Here $p_0(w)$ is the translation probability of word w which we define as:

$$p_0(w) = \begin{cases} \frac{1}{|Q|} * \frac{1}{2} & \text{if } w \in L_1 \text{ and } w \in Q \\ 0 & \text{if } w \in L_1 \text{ and } w \notin Q \\ \sum_{q \in Q} p(w|q) * \frac{1}{2} & \text{if } w \in L_2 \end{cases}$$

$p_{MI}(w_i \rightarrow w_j)$ is the normalized weight of the co-occurrence edges between words in the same language and $p_{trans}(w_i \rightarrow w_j)$ is the translation probability. α_0, α_{MI} and α_{trans} control the influence of each component on the total score of each word.

These probability scores are computed iteratively, updating the probability score of each word using the updated probability scores of the neighbors until they converge to a limit. We then estimate the query language model in L_2 by normalizing these probability scores:

$$p(w|\hat{\Theta}_Q) = \frac{p(w)}{\sum_{v \in L_2} p(v)} \quad \text{for each } w \in L_2$$

Using this updating formula, at each step, the score of each word is propagated to its outgoing neighbors in the word network in a weighted manner, and the score of each word is updated to a combination of the sum of its incoming (propagated) scores and its own score.

The score definitions 5.1 and 5.2 correspond to the standing probability distribution of a random walk on the word network. Think of a random surfer surfing the set of words looking for terms related to the query Q. At each step, the surfer being in a word, jumps to a co-occurring word with probability α_{MI}, jumps to a correlated word with probability α_{trans} and jumps to a random word based on its translation probability with probability α_0. The surfer keeps doing this iteratively, jumping to words looking for words related to the query. The final score of each word is equal to the stationary probability of the surfer visiting the word.

In order to compute the scores, we construct a matrix $M = \alpha_0 M_0 + \alpha_{MI} M_{MI} + \alpha_{trans} M_{trans}$. Here $M_0(m,n) = \beta p_0(w_n) + (1-\beta)/|N|$ where β is a number very close to 1 (0.99 in our experiments) which is used to give unreachable words a very tiny probability. $M_{MI}(m,n) = p_{MI}(w_m \rightarrow w_n)$ and $M_{trans}(m,n) = p_{trans}(w_m \rightarrow w_n)$. We then compute the probability scores using matrix multiplication: $\overrightarrow{P} = M^T \overrightarrow{P}$ where $\overrightarrow{P}$ is the vector of the probability values. The probability scores are computed iteratively until they converge to a unique probability distribution. Clearly, efficient matrix multiplication methods can be used to further speed up the scoring. The final scores will be the values of the stationary probability distribution of the Markov chain defined by M. The way we have defined M_0 will ensure reachability to each word, thus by the Ergodicity theorem for Markov chains, we know that the Markov chain defined by such a transition matrix M must have a unique stationary probability distribution.

Connection to the General Probabilistic Score Propagation Framework

The proposed propagation framework is clearly a special case of the general probabilistic propagation framework proposed in Chapter 2. Here the words in the two languages are the nodes of the network and the neighbor sets are co-occurring words in the same language and correlated words in different languages.

We can easily rewrite the updating equations 5.1 and 5.2 as:

$$\begin{aligned} p(w^1) = \alpha_0 & \sum_{x \in V} p(x) p_0(x \rightarrow w^1) \\ & + \alpha_{MI} \sum_{x \in L_1} p(x) p_{MI}(x \rightarrow w^1) \\ & + \alpha_{trans} \sum_{y \in L_2} p(y) p_{trans}(y \rightarrow w^1) \qquad \text{if } w^1 \in L_1 \end{aligned}$$

and

$$
\begin{aligned}
p(w^2) = \alpha_0 \quad & \sum_{x \in V} p(x) p_0(x \rightarrow w^2) \\
& + \alpha_{trans} \sum_{x \in L_1} p(x) p_{trans}(x \rightarrow w^2) \\
& + \alpha_{MI} \sum_{y \in L_2} p(y) p_{MI}(y \rightarrow w^2) \qquad \text{if } w^2 \in L_2
\end{aligned}
$$

or put them together as:

$$
\begin{aligned}
p(w) = \alpha_0 \quad & \sum_{x \in V} p(x) p_0(x \rightarrow w) \\
& + \alpha_{MI} \sum_{x \in V} p(x) p_{MI}(x \rightarrow w) \\
& + \alpha_{trans} \sum_{x \in V} p(y) p_{trans}(x \rightarrow w)
\end{aligned}
$$

where $\alpha_0 + \alpha_{MI} + \alpha_{trans} = 1$, V is the set of all words in L_1 and L_2, $p_0(x \rightarrow w) = p^0(w)$ and

$$
\begin{aligned}
p_{MI}(x \rightarrow w) > 0 \quad & \text{if } x, w \in L_1 \text{ or } x, w \in L_2 \\
= 0 \quad & \text{o.w.}
\end{aligned}
$$

$$
\begin{aligned}
p_{trans}(x \rightarrow w) > 0 \quad & \text{if } x \in L_1, w \in L_2 \text{ or } x \in L_2, w \in L_1 \\
= 0 \quad & \text{o.w.}
\end{aligned}
$$

The rewritten updating formula is obviously a special case of the updating propagation equation 2.1.

Ranking Documents

Having generated the query language model in L_2, we then rank the documents in L_2 based on the KL-divergence between the estimated query language model and the estimated document language models [101]. We assume that each document is generated from a unigram document language

model Θ_D. We estimate the document language model ($\hat{\Theta}_D$) of each document using the maximum likelihood estimator and smooth the estimated document language models using Dirichlet prior smoothing.

Assuming $\hat{\Theta}_Q$ and $\hat{\Theta}_D$ to be the estimated query and document language models respectively, the document D is ranked based on the KL-divergence between $\hat{\Theta}_Q$ and $\hat{\Theta}_D$:

$$-D(\hat{\Theta}_Q||\hat{\Theta}_D) = -\sum_{w \in V} p(w|\hat{\Theta}_Q) log \frac{p(w|\hat{\Theta}_Q)}{p(w|\hat{\Theta}_D)}$$

where V is the set of words in our vocabulary.

5.4 Experiments

In our experiments, we focus on cases at which we do not have rich linguistic resources such as bilingual dictionaries or machine translation systems. All we have is comparable bilingual corpora, which are widely available on the Web for different languages. We will show that with such limited linguistic resources and compared to the monolingual baseline, we can achieve up to 75.9% of mean average precision, up to 76.5% of precision at 5 documents and up to 77.2% of precision at 10 documents.

5.4.1 Data Set and Queries

As the comparable corpora, we used Arabic-English comparable corpora from news articles published by Agence France Presse and Xinhua news agencies. They are parts of the Arabic and English Gigaword corpora. The articles are aligned based on the date of publication.

As the cross-language information retrieval task we focus on the CLIR task of TREC-2002 [54]: Retrieval of Arabic documents from topics in English. The document collection for this task contains 383,872 newswire stories (896MB) that appeared on the Agence Franaise de Presse (AFP) Arabic Newswire between 1994 and 2000. The queries are 50 topic descriptions in English and the

Table 5.1: Monolingual Arabic-Arabic retrieval performance

Run	MAP	Prec@5	Prec@10
No Query Expansion	0.2791	0.396	0.398
Query Expansion	0.3449	0.476	0.438

Table 5.2: Title-only monolingual performance of TREC-2002 teams

Team	MAP	Prec@5	Prec@10
Hummingbird Technologies	0.2782	0.4000	0.3840
IBM Research	0.3030	0.3800	0.3960
University of Neuchatel	0.3572	0.5040	0.4580

Arabic translations of these topics. The Arabic translations are used for the monolingual retrieval.

5.4.2 Monolingual (Arabic-Arabic) Retrieval

We use monolingual Arabic-Arabic retrieval as a baseline to which we compare the cross-language results. In our monolingual Arabic runs, we only use the title field of each Arabic query topic as the query words. We used the light10 Arabic stemmer in the Lemur toolkit to stem the Arabic words. This light stemmer strips off initials, definite articles and suffixes. We did our experiments with two versions of the monolingual run: one that does not do query expansion, and one which does query expansion with pseudo-feedback. For the pseudo-feedback runs, we used the top 10 retrieved documents to perform feedback using the mixture model approach implemented in the Lemur toolkit [101]. As the parameters, we used 0.5 for both background noise and feedback coefficient and used 100 terms for expanding the query model. Table 5.1 shows the pean average precision, precision at 5 documents and precision at 10 documents of our monolingual runs.

Among the nine teams participating in the TREC-2002 cross-language information retrieval track, three did monolingual Arabic retrieval with title fields only: Hummingbird Technologies [89], IBM Research [20] and University of Neuchatel [77]. Table 5.2 shows the title-only monolingual results of these three runs. All these teams use blind query expansion for these monolingual runs. Our monolingual results with query expansion is better than the results obtained by Hummingbird

Technologies and IBM Research, but slightly worse than the results of University of Neuchatel. As the tables show, our results are comparable to these monolingual results and form a reasonable baseline to which we can compare our cross-language results.

5.4.3 Naive Probability Estimation

In our first set of experiments, we first construct the word mappings between all the English query words and their possible Arabic translations. As the possible Arabic translations, for each English query word, we consider all the Arabic words occurring in any Arabic document time-aligned with the English document the English query term has occurred in. We further prune those Arabic words which occur very frequently and those with very low frequency.

From the obtained word correlations, we select those above a specific threshold with the intuition that these correlations are more reliable. We then use the naive probability estimation method to estimate translation probabilities of English and Arabic words from the raw correlation scores:

$$p(u_i|w) = \frac{r_i}{\sum_{j=1}^{N} r_j}$$

Having computed the translation probabilities, we then use each of our proposed methods to construct the corresponding Arabic query language model of the English queries and rank the documents based on the KL-divergence between the estimated query language models and the document language models.

Basic Query Translation

In this set of experiments, we used our proposed Top-K translation method to construct the Arabic query language model corresponding to each English query using the estimated probabilities and used these Arabic queries to retrieve Arabic documents. Tables 5.3 (a) and 5.3 (b) show the results of Top-2 to Top-10 translations, where we use the top 2 to top 10 correlated words as the translations of each query word, when we do not expand queries and when we do query expansion with

pseudo feedback respectively. In this set of experiments, we have set the correlation threshold to 0.5, only counting words with correlation scores above this threshold as translations of the query words.

Table 5.3: Basic query translation and Naive probability estimation

(a) No Query Expansion

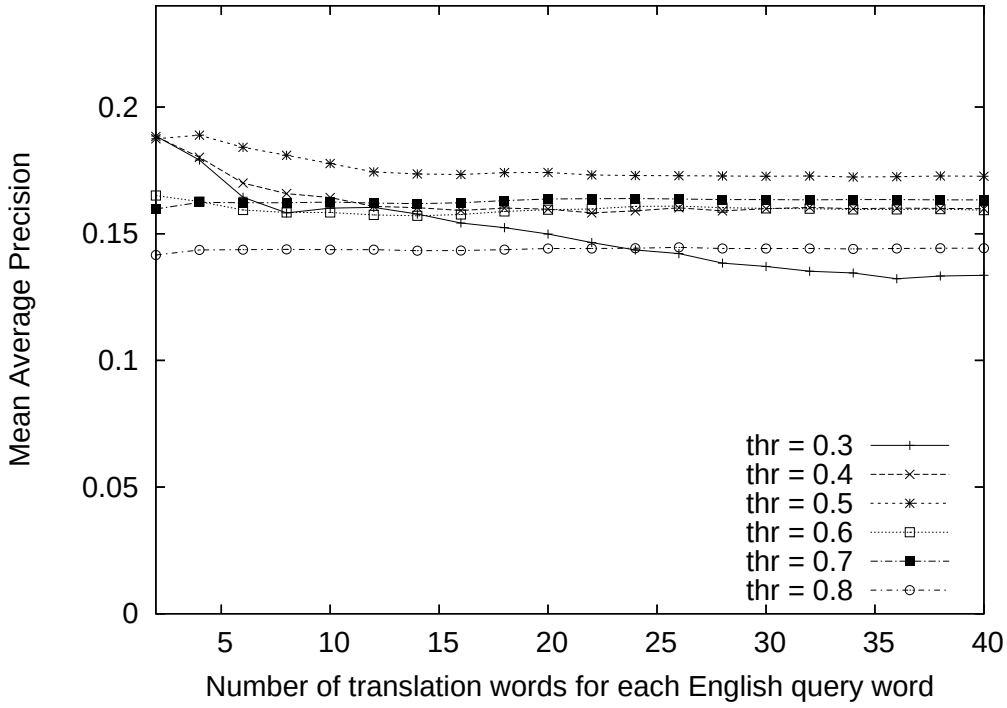
Method	MAP	% of Mono	Prec@5	% of Mono	Prec@10	% of Mono
Mono Baseline	0.2791		0.396		0.398	
Top-2	0.1875	67.2%	0.2939	74.2%	0.2816	70.8%
Top-4	0.1889	67.7%	0.3061	77.3%	0.2796	70.3%
Top-6	0.1841	66%	0.2694	68%	0.2633	66.2%
Top-8	0.181	64.9%	0.2694	68%	0.2694	67.7%
Top-10	0.1777	63.7%	0.2571	64.9%	0.2653	66.7%

(b) Query Expansion with pseudo feedback

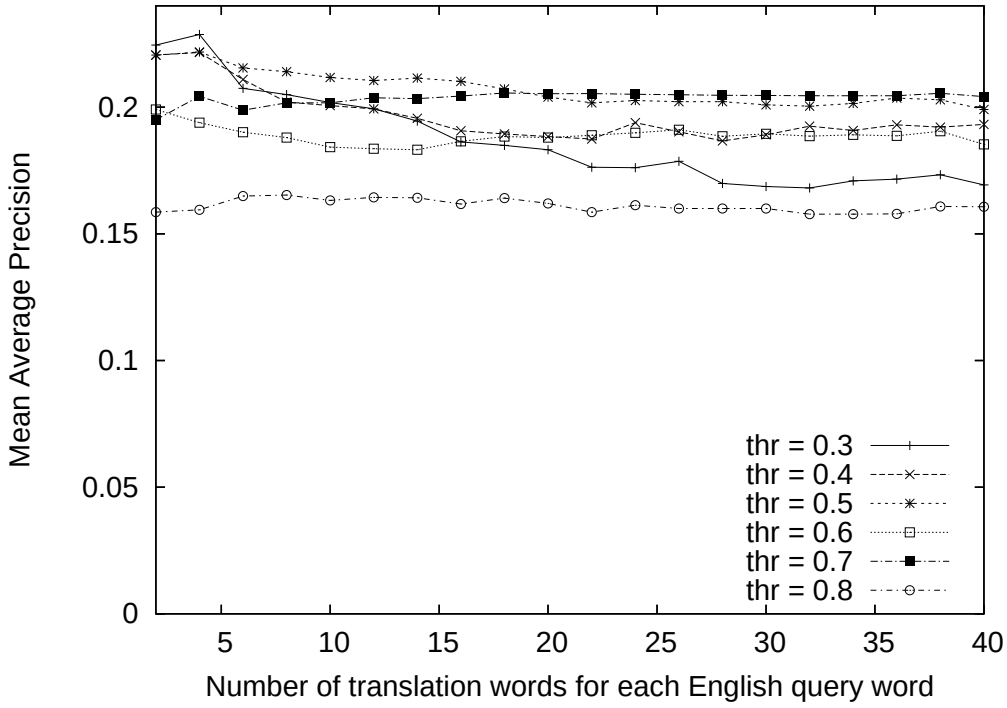
Method	MAP	% of Mono	Prec@5	% of Mono	Prec@10	% of Mono
Mono Baseline	0.3449		0.476		0.438	
Top-2	0.2206	64%	0.3061	64.3%	0.3041	69.4%
Top-4	0.2218	64.3%	0.3224	67.7%	0.3041	69.4%
Top-6	0.2155	62.5%	0.2898	60.9%	0.2878	65.7%
Top-8	0.214	62%	0.2857	60%	0.2796	63.8%
Top-10	0.2117	61.4%	0.2694	56.6%	0.2673	61%

As the tables show, using this naive probability estimation and basic query translation and compared to the monolingual baseline, we can achieve up to 67.7% of mean average precision, 77.3% of precision at 5 documents and 70.8% of precision at 10 documents when not doing query expansion and about 64.3% of mean average precision, 67.7% of precision at 5 and 69.4% of precision at 10 documents when we expand queries. These results are very promising as we are using very little language resources for this task.

As we stated earlier, we prune those Arabic words with correlation scores bellow the threshold to eliminate unreliable translations. We further tried to change the value of this threshold to see how it affects the performance. Figures 5.5 (a) and 5.5 (b) show the mean average precision for different number of translation words as we vary the threshold from 0.3 to 0.8, when we do not



(a) No query expansion



(b) Query expansion with pseudo feedback

Figure 5.5: Using different thresholds for pruning Arabic translations

expand the queries and when we do query expansion with pseudo feedback respectively.

When we set the threshold to 0.3, we almost allow all correlated words, even with small correlation scores, to be counted as translation words. Thus increasing the number of translation words for each English query word will allow inaccurate translation words to enter the query translation and thus hurt the performance. As we increase the value of the threshold, we prune those unreliable translation words, thus increasing the number of translation words will not hurt the performance as much.

With no query expansion, we get the best performance when we set the threshold to 0.5. After that, increasing the value of threshold will decrease the mean average precision. That is because we are setting the threshold too tight and we have very few translation words for our English query words. This is different in the case when we do query expansion. As can be seen in the figure, a high threshold such as 0.7 results in a high mean average precision. This can be because with such a high threshold, only few accurate translation words remain which results accurate relevant documents on top. We use these top documents for blind query expansion, which leads to better performance. Again increasing the value of the threshold to 0.8 hurts the performance, because of pruning most of correlated words which results no translation words for many of our English query words.

Query Translation using Propagation Method

In our next set of experiments, we used our proposed propagation method with the hope to better estimate the query language models. Recall that using this method, we consider both word co-occurrences and word correlations when constructing the Arabic query language models. Tables 5.4 (a) and 5.4 (b) show the results of this set of experiments when we do not expand queries and when we do query expansion respectively. In these tables, we report mean average precision, precision at 5 documents and precision at 10 documents when we use the top 2 to top 10 translation words as correlation neighbors. The tables also show the improvement we get using the propagation method over the basic translation method.

Table 5.4: Query translation using propagation and Naive probability estimation

(a) No query expansion

# of translation neighbors	MAP	Impr.	Prec@5	Impr.	Prec@10	Impr.
2	0.2033	8.4%	0.332	13%	0.3102	10.2%
4	0.1981	4.9%	0.298	-	0.2898	3.6%
6	0.1925	4.6%	0.2612	-	0.272	3.3%
8	0.1902	5.1%	0.2735	1.5%	0.276	2.4%
10	0.1876	5.6%	0.268	4.2%	0.2653	-

(b) Query expansion with pseudo feedback

# of translation neighbors	MAP	Impr.	Prec@5	Impr.	Prec@10	Impr.
2	0.235	6.5%	0.3429	12%	0.3245	6.7%
4	0.2271	2.4%	0.3143	-	0.3	-
6	0.2229	3.4%	0.2939	1.4%	0.2939	2.1%
8	0.221	3.3%	0.2939	2.9%	0.2898	3.6%
10	0.2189	3.4%	0.28	3.9%	0.2776	3.9%

As the tables show, both when we do not expand queries and when we do query expansion, we can improve the performance in most cases using our propagation method. We get the best performance when we use the top 2 translation words as correlation neighbors.

From Table 5.4 (a), it is clear that we can improve the performance a lot when we use the top 2 translation words as correlation neighbors and when do propagation for constructing the query language models and we do not expand the queries. The reason is that the propagation framework gives us some kind of feedback effect itself, allowing co-occurring words to appear in the query translation and affect the estimated probabilities. Compared to the monolingual baseline when we do not expand queries, we get up to 72.8% of mean average precision, up to 83.8% of precision at 10 documents and up to 77.9% of precision at 10 documents. We also see improvement using propagation when we expand queries. Compared to the monolingual baseline when we expand queries, we get up to 68.1% of the mean average precision, up to 72% of precision at 5 and up to 74.1% of precision at 10.

5.4.4 Exponential Transformation of the Correlations

In the next set of experiments, we used exponential transformation of correlations to penalize low correlations, which we assume are unreliable, and estimated the translation probabilities from the transformed correlation scores. Recall that the transformation function we used was:

$$f(r) = \frac{1}{e^b - 1} e^{br} - \frac{1}{e^b - 1}$$

These probabilities drop sharply as the correlation scores become smaller. We did the experiments again with these new probability scores.

Basic Query Translation

In this set of experiments, we used our basic query translation method to construct the Arabic queries again, but using the new transformed probability scores. As we expected, exponential transformation of the correlations helped us improve the results significantly. Tables 5.5 (a) and 5.5 (b) show two sample sets of results with $b = 6$ and $thr = 0.3$.

We further tried different values of b, the parameter of the exponential transformation which controls the amount we want to penalize low correlations, to see how the performance changes. Figures 5.6 (a) and 5.6 (b) show the performance for different values of b. The horizontal axis shows the number of translation words we use for each query word and the vertical axis shows the mean average precision. In these set of experiments, we set the correlation threshold to 0.3.

As the charts show, exponential transformation improves the performance over the naive probability estimation baseline with all the different values of b, showing that it is a reasonable transformation when computing the probabilities. We get the best performance at $b = 8$, were compared to the monolingual baseline, we get up to 76% of the mean average precision with no query expansion and up to 72.2% of mean average precision with query expansion.

One interesting observation is that as we increase the number of translation words, our baseline performance with no exponential transformation hurts a lot. But with our exponential transforma-

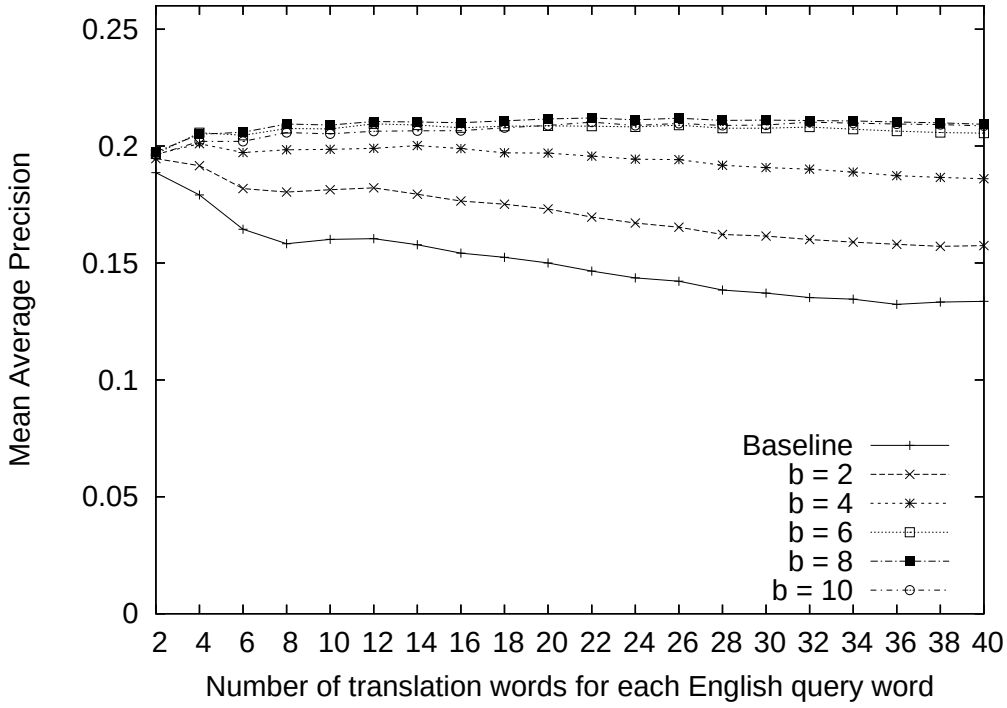
Table 5.5: Basic query translation and Exponential transformation of the correlations (b = 6, threshold = 0.3)

(a) No query expansion

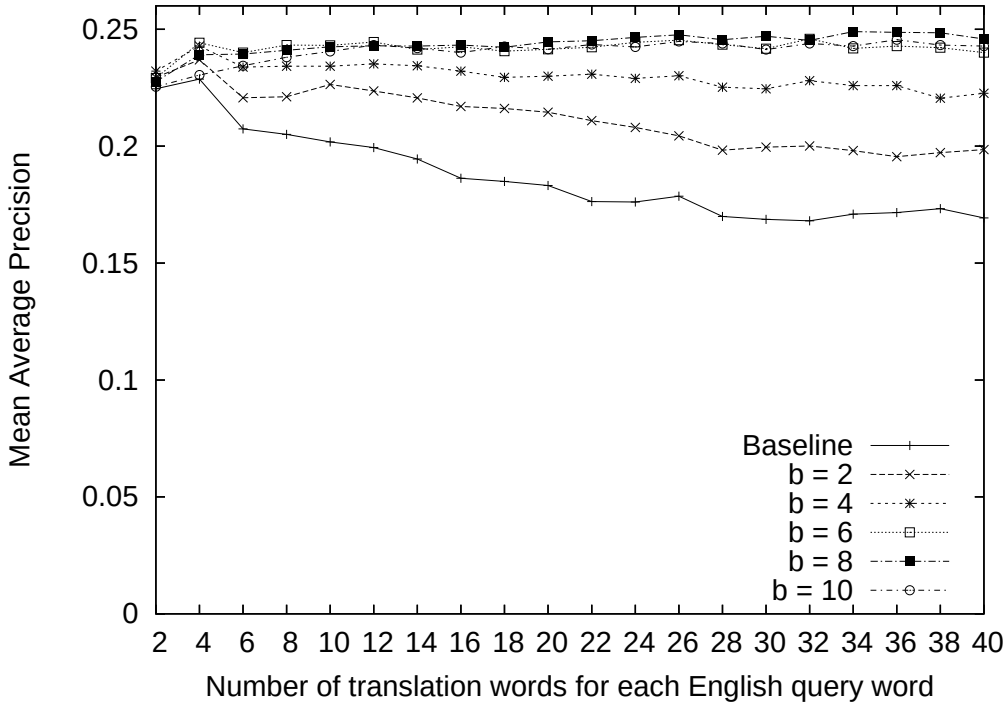
Method	MAP	% of Mono	Impr.	Prec@5	% of Mono	Impr.	Prec@10	% of Mono	Impr.
Mono	0.2791			0.396			0.398		
Top-2	0.1966	70.4%	4.2%	0.292	73.7%	2.8%	0.292	73.4%	-
Top-4	0.2057	73.7%	14.9%	0.328	82.8%	1.2%	0.302	75.9%	4.9%
Top-6	0.2046	73.3%	24.4%	0.3	75.8%	13.6%	0.302	75.9%	16.2%
Top-8	0.2075	74.3%	31.2%	0.316	79.8%	21.5%	0.292	73.4%	16.8%
Top-10	0.2073	74.3%	29.5%	0.316	79.8%	16.2%	0.298	74.9%	18.3%

(b) Query expansion with pseudo feedback

Method	MAP	% of Mono	Impr.	Prec@5	% of Mono	Impr.	Prec@10	% of Mono	Impr.
Mono	0.3449			0.476			0.438		
Top-2	0.2291	66.4%	2%	0.304	63.%	1.3%	0.3	68.5%	-
Top-4	0.2442	70.8%	6.8%	0.336	70.6%	1.2%	0.33	75.3%	1.9%
Top-6	0.24	69.6%	15.7%	0.312	65.5%	14.7%	0.31	70.8%	6.2%
Top-8	0.2432	70.5%	18.6%	0.328	68.9%	20.6%	0.312	71.2%	9.9%
Top-10	0.2431	70.5%	20.5%	0.324	68.1%	14.1%	0.306	69.9%	10.1%



(a) No query expansion



(b) Query expansion with pseudo feedback

Figure 5.6: Exponential transformation with different values of b (threshold = 0.3)

tion, specially with larger values of b, the performance improves and stabilizes at some point. This is exactly what we expected. With our exponential transformation, increasing the number of translation words will not allow inaccurate translation words to hurt the performance, since they only have very small probabilities.

In this set of experiments, we had fixed the threshold to 0.3. We further tried different thresholds for our best set of results ($b = 8$) to see how it affects the performance. Figures 5.7 (a) and 5.7 (b) show the performance when we change the threshold from 0.3 to 0.8.

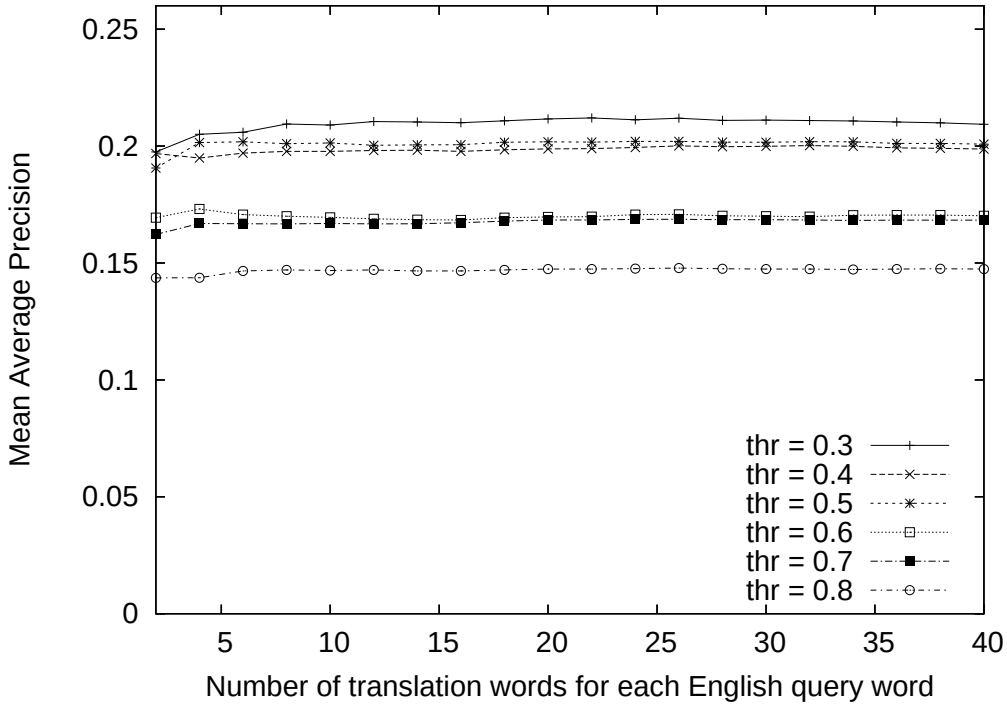
The performance change this time is different from what we saw when we used the correlation values directly. When we use exponential transformation to compute the probabilities, we are penalizing those words with small correlation values, thus increasing the number of translation words does not hurt the performance. Besides, we get the best performance at $thr = 0.3$, that is when we use most correlated words. Setting higher thresholds prune some correct translation words which would allow better performance if we had kept them, even with small probability values.

Figures 5.8 (a) and 5.8 (b) show the performance for different values of b when we set the threshold to 0.5.

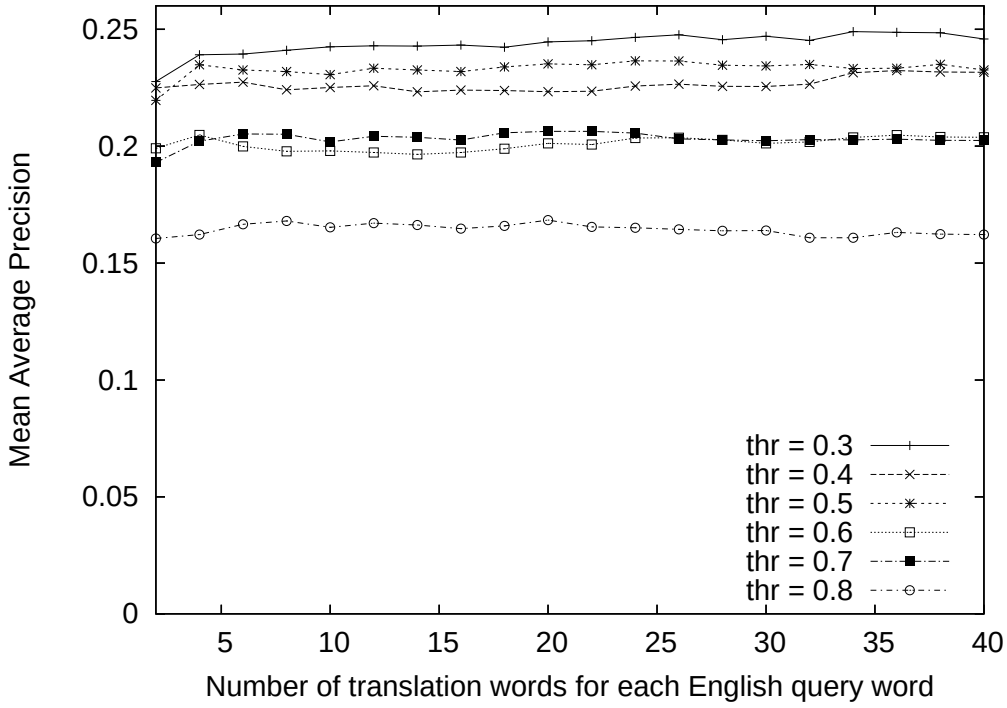
As can be seen from the charts, the best performance in this case is worse than the case when we set the threshold to 0.3. That is because we are pruning some effective translation words when we set a higher threshold. Otherwise the trend is similar.

Query Translation using Propagation Method

We finally ran the experiments using propagation method and the new transformed probability scores. Tables 5.6 (a) and 5.6 (b) show the results of this set of experiments when we do not expand queries and when we do query expansion respectively. In these tables, we report the performance results when we use the top 2 to top 10 translation words as correlation neighbors. We also show the improvement we get using the propagation method over the basic translation method in these tables.

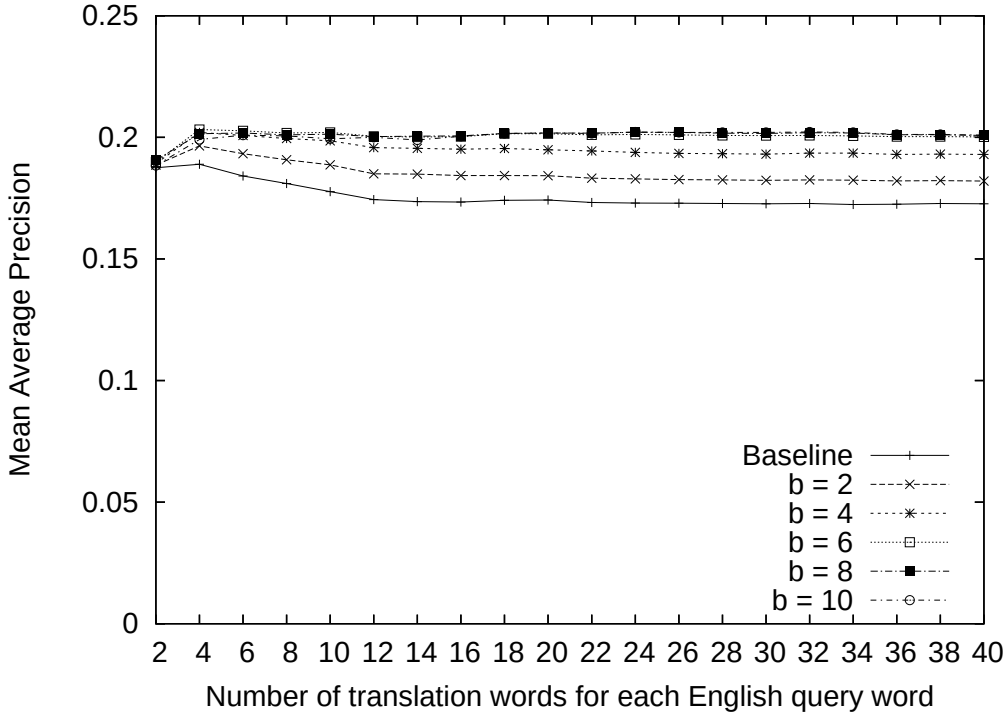


(a) No query expansion

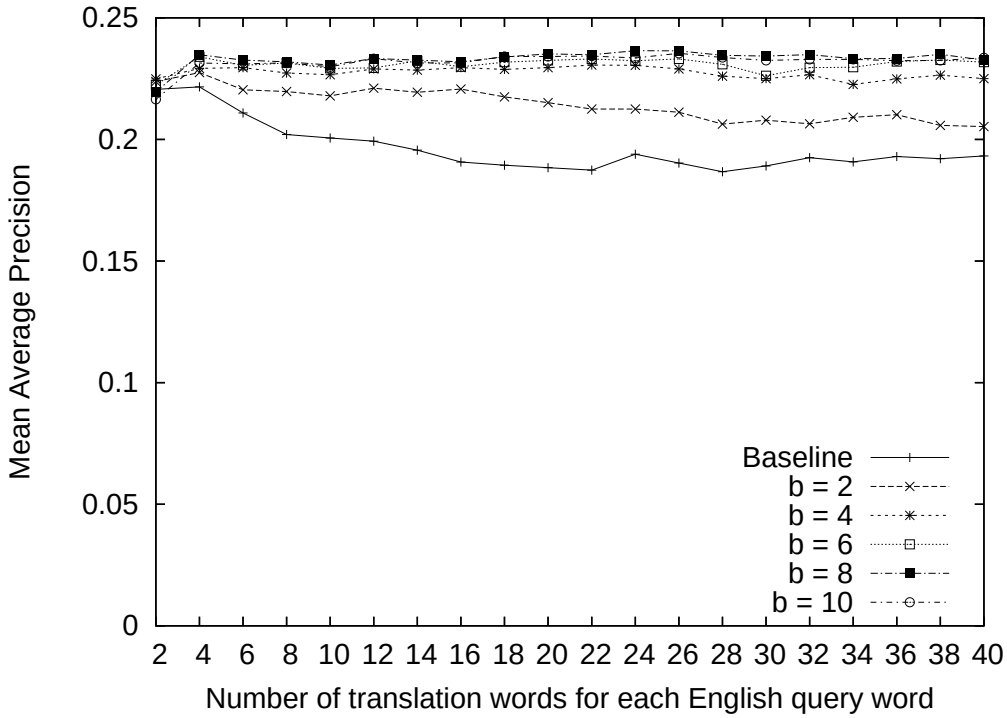


(b) Query expansion with pseudo feedback

Figure 5.7: Exponential transformation with different thresholds (b = 8)



(a) Query expansion



(b) No query expansion

Figure 5.8: Exponential transformation with different values of b (threshold = 0.5)

Table 5.6: Query translation using propagation and Exponential transformation of correlations

(a) No query expansion

# of translation neighbors	MAP	Impr.	Prec@5	Impr.	Prec@10	Impr.
2	0.2129	8.3%	0.32	9.6%	0.318	8.9%
4	0.2259	9.8%	0.336	2.4%	0.316	4.6%
6	0.2239	9.4%	0.336	12%	0.332	9.9%
8	0.2212	6.6%	0.344	8.9%	0.326	11.6%
10	0.2242	8.2%	0.34	7.6%	0.324	8.7%

(b) Query expansion with pseudo feedback

# of translation neighbors	MAP	Impr.	Prec@5	Impr.	Prec@10	Impr.
2	0.2428	6%	0.332	9.2%	0.336	12%
4	0.2617	7.2%	0.348	3.6%	0.334	1.2%
6	0.2579	7.5%	0.356	14.1%	0.338	9%
8	0.2561	5.3%	0.364	11%	0.336	7.7%
10	0.2549	4.9%	0.352	8.6%	0.328	7.2%

As the tables show, in both cases, when we do not expand queries and when we do query expansion with pseudo feedback, we get significant improvements using our propagation method over the basic method. In the case when we do not expand queries and compared to the monolingual baseline, we achieve up to 80.3% of mean average precision, up to 86.9% of precision at 5 documents and up to 83.4% of precision at 10 documents. When we expand queries, we achieve up to 74.8% of mean average precision, up to 76.5% of precision at 10 documents and up to 77.2% of precision at 10 documents.

Sensitivity Analysis

We have so far reported the best performance we achieve through tuning the parameters α_{MI} and α_{trans} in the propagation model which control the influence of each group of neighbors. The question now is how sensitive this method is to the setting of these parameters.

To answer this question, we compute the range of parameter values for which using propaga-

tion outperforms the baseline method. Figures 5.9 and 5.10 show the performance results with different parameter values for the propagation method. The dark gray cells indicate the case where propagation outperforms the baseline, light gray cells indicate equal performance and white cells indicate the case where the baseline outperforms the propagation method. It can easily be seen that the optimal range is quite wide. Specifically, the propagation method improves the mean average precision for all the values of the parameters. precision at 5 documents and precision at 10 documents are more sensitive to the setting of these parameters, but still for a wide range of parameter values, the propagation method outperforms the baseline method, indicating that using the propagation method is useful in improving the performance in general.

5.5 Summary

Existing work on cross-language information retrieval has mostly relied on rich, high quality linguistic resources such as machine translation systems, bilingual dictionaries, or parallel corpora. But such high quality resources often do not exist for many minority language pairs, making it a challenge to perform cross-language IR for such language pairs. We observed that for these language pairs, we often naturally have available comparable corpora, and studied how to use just comparable corpora to do cross-language information retrieval. Our basic idea is to use word associations extracted from comparable corpora based on time correlations to translate queries from the source language to the target language. In order to improve the estimation of the target query language model, we propose an exponential transformation function to increase the robustness of term weighting. We further propose a probabilistic propagation framework which exploits word co-occurrences in the monolingual data as well to better estimate the query language model.

The experiment results show that it is feasible to use just comparable corpora to do CLIR and using the proposed transformation function can achieve up to 70.8% of mean average precision, 70.6% of precision at 5 documents and 75.3% of precision at 10 documents compared to the monolingual retrieval performance. We further observed that the propagation method is and effective

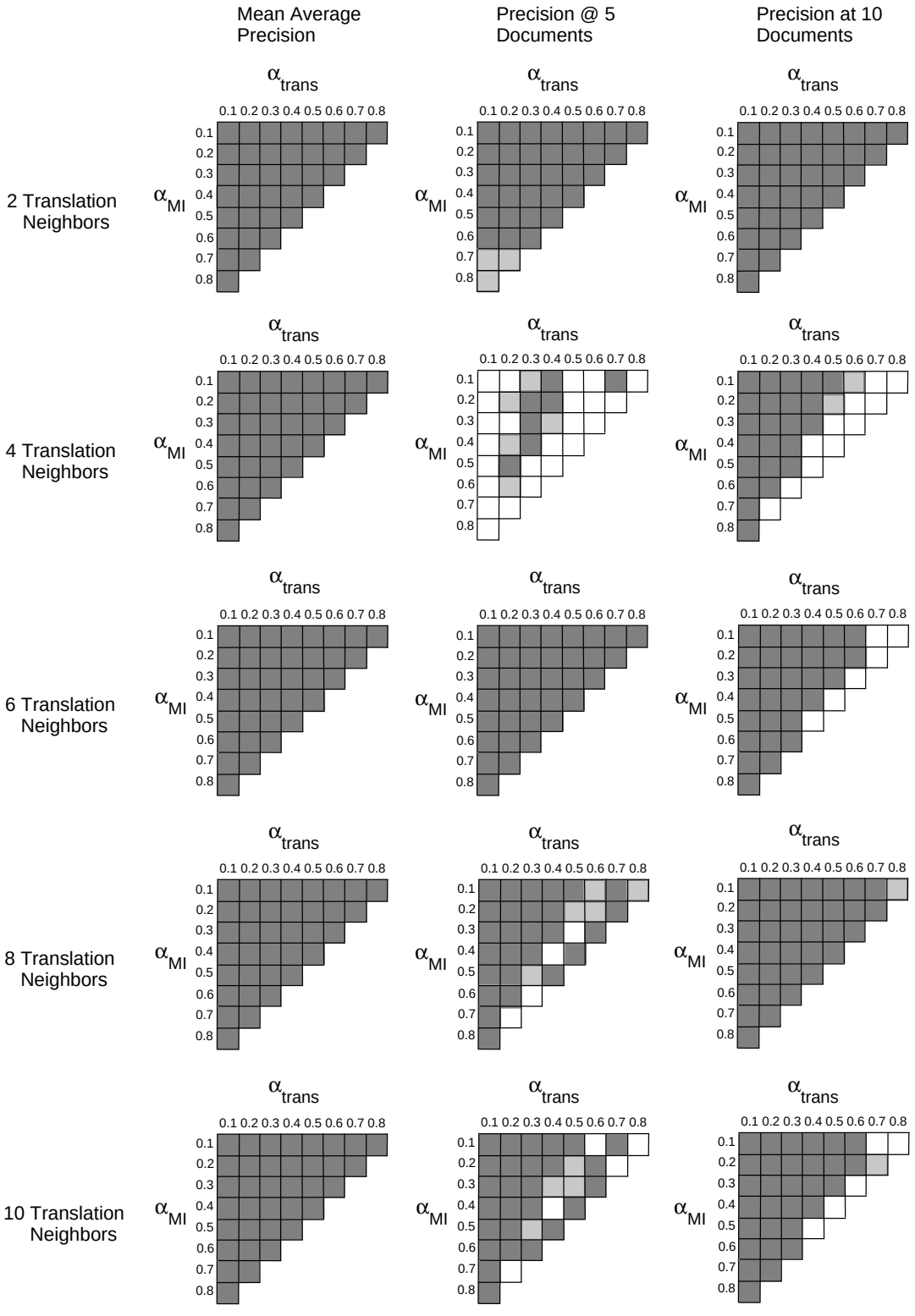


Figure 5.9: Ranges of α_{MI} and α_{trans} for improving baseline - No query expansion

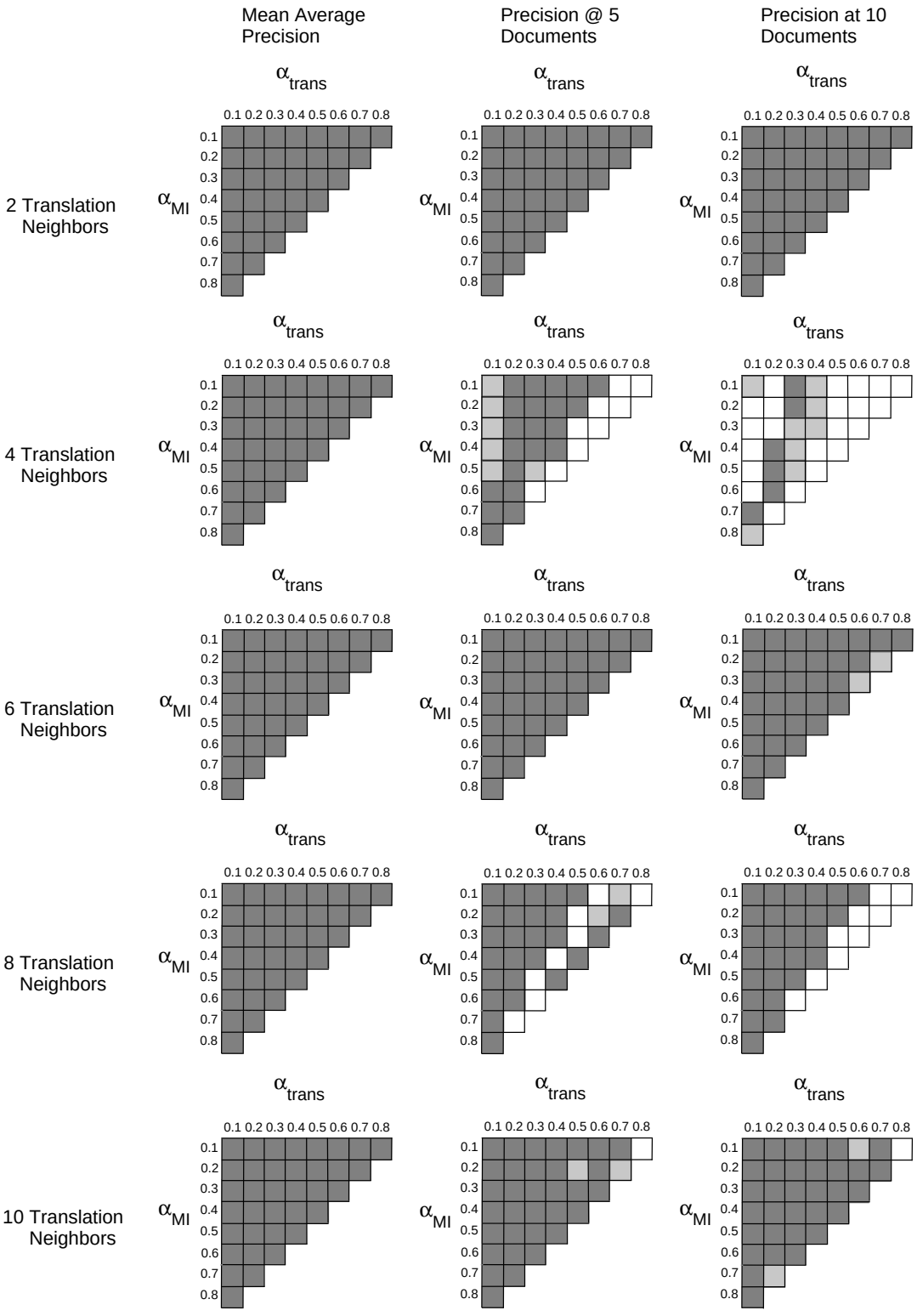


Figure 5.10: Ranges of α_{MI} and α_{trans} for improving baseline - Query expansion with pseudo feedback

method which can improve the performance substantially, achieving up to 75.9% of mean average precision, 76.5% of precision at 5 documents and 77.2% of precision at 10 documents. These results are quite promising since we are using very limited naturally available linguistic resources, thus the method can potentially be applied to do CLIR for many minority language pairs.

There are several interesting directions for further research in this area:

1. In our current experiment setup, the queries and the comparable corpora are from the same domain. We currently use news articles published in Arabic and English languages as our comparable corpora and our cross-language information retrieval task is to retrieve Arabic newswire stories in response to English queries. Intuitively, if the query and the comparable corpora are not from the same domain, the CLIR task should be harder. An interesting future research direction is to look into cases where queries are slightly out of the domain of the comparable corpora to see how our method performs. Specifically, we should look into the coverage of query words in the comparable corpora and its impact on the performance. We can also think of other sources for comparable corpora, for example scientific literatures to extract translation knowledge.

2. We are currently using comparable corpora as the only available language resource, but potentially our method can benefit CLIR methods which use other linguistic resources such as bilingual dictionaries. Exploiting comparable corpora can be expected to help when the vocabulary coverage of common dictionaries is poor for a language. A good example of this is Germany's top selling newspaper Bild. Many of the words in this newspaper are not covered in any of the common German dictionaries. These are the words that are newly introduced to the language. Using the comparable corpora on top of the bilingual dictionaries can help to find translations of these new words.

3. A different solution to the cross-language information retrieval problem involving a resource-lean language pair (e.g. Arabic-Lithuanian) for which we do not have rich linguistic resources is to go through a third popular language. For example for translating a query from

Arabic to Lithuanian, we can first translate the Arabic query to English and then translate the English translation to Lithuanian (Assuming we have linguistic resources for Arabic-English and English-Lithuanian language pairs). An interesting research direction is how to use our propagation framework for such a double translation approach.

Chapter 6

Conclusions

6.1 Summary

Applications of information retrieval deal with different units of information which are connected through explicit or implicit link structures. For example a hypertext collection is composed of documents connected through different kinds of links, such as explicit hyperlinks or implicit co-citation links. Principled combination of these two sources of information, namely content information of units and link structure, is critical for achieving good retrieval performance. Traditional information retrieval methods either only use the content information for retrieval and ignore the link structure or not fully exploit the discrimination power of contents as well as all useful link information.

In this thesis, we have proposed a general probabilistic score propagation framework for combining content and link information which can fully take advantage of content information and the link structure in a principled way. The proposed framework takes a strict probabilistic view on the score propagation model, making the weights of propagation meaningful and providing guidance on how to normalize content scores and how to set propagation parameters to optimize retrieval accuracy. Another characteristic of the propagation framework is propagating through multiple groups of neighbors which is shown to outperform the results of using a single type of neighbors.

We have studied three different applications of the proposed framework in this thesis.

As the first application, we applied the general probabilistic score propagation framework to a hypertext collection expanded with implicit links between documents to generate a probabilistic relevance propagation framework for hypertext retrieval. We showed that the generated framework

can unify most existing link-based ranking algorithms and can also suggest several interesting new algorithms through different propagation strategies. Using the generated framework, we systematically compared eight different relevance propagation models on two TREC test collections. The experiment results show that all the eight relevance propagation models outperform the baseline content-only method for a wide range of values, indicating that the generated probabilistic relevance propagation framework provides a general, effective and robust way of exploiting link information to improve hypertext search accuracy. The experiment results also show that using multiple groups of neighbors for propagation outperform using just one type of neighbor and that the strict probabilistic view of propagation provides guidance on setting propagation parameters.

In the second application, we focused on using the proposed general framework to do smoothing of document language models in language modeling approaches to information retrieval. In this application, we cast the problem of smoothing document language models as a problem of propagating term counts among documents probabilistically. We applied the general probabilistic score propagation framework to the graph of documents with generation links and came up with the probabilistic term count propagation algorithm and presented a novel method of smoothing document language models using this term propagation algorithm. A major characteristic of the proposed method is that it provides a principled way to bring in remotely related documents to smooth the current document. The experiment results show that the proposed method significantly outperforms the simple collection-based smoothing method and smoothing with remote neighbors in the document similarity graph outperforms smoothing with only immediate neighbors. Compared with other smoothing methods that also exploit local corpus structures, this method is especially effective in improving precision in top-ranked documents through filling in missing query terms in relevant documents, which is presumably most important in practical applications as a user often only reads a few top-ranked documents. Furthermore, the experiment results show that this method is complementary with pseudo feedback which tends to improve the average precision, and a combination of the two methods achieves better performance than either one alone.

Our third application was applying the general framework to do cross-language information

retrieval. In this application, we focused on cases where we do not have rich linguistic resources between language pairs. All we have is comparable corpora which are often naturally available. For this application, we applied the general probabilistic score propagation framework to a graph of terms with implicit mutual information and correlation links to generate a probabilistic score propagation model for cross-language information retrieval. With the generated model, we iteratively propagated term statistics in the term graph to construct the target query language model corresponding to the given query. We then used these query language models to retrieve documents in the target language. The experiment results show that the proposed method is effective for this task. Specifically, compared to the monolingual baseline results, we can achieve up to 75.9% of mean average precision, 76.5% of precision at 5 documents and 77.2% of precision at 10 documents when we use the proposed method. These results are very promising since we are using very limited naturally available linguistic resources. This method can potentially be used to do cross-language information retrieval in many minority language pairs.

The proposed framework is a very general framework that can be applied to diverse applications of information retrieval. In this thesis, we studied three different applications of this general framework in three distinct areas of information retrieval and showed that it provides a general effective way of exploiting content and link information to improve retrieval accuracy. These three applications are only a few samples of potentially many applications that can benefit from this general framework.

6.2 Future Directions

There are many possible future directions of research on this topic.

New Applications

The proposed propagation framework is a general framework that has potential applications in different areas of information retrieval. We can further study applying the framework on other

new problems. One application that can potentially benefit from the framework is Multi-Lingual Information Retrieval (MLIR). Multi-lingual information retrieval concerns the task of satisfying a query with documents in multiple different languages, which can be though of as an aggregation of a set of CLIR tasks in a certain degree. Thus we can possibly extend the method proposed for cross-language information retrieval for doing MLIR.

Another possible application is XML retrieval. Extensible markup language (XML) has a widespread use in scientific data repositories, digital libraries and the world wide web. Thus there has been a lot of research on developing techniques for XML retrieval. In these techniques, the logical structure of the documents, explicitly represented by XML markup, is used to retrieve document components. Intuitively, the XML retrieval task perfectly fits our proposed framework. Given a set of XML documents, we can construct a graph composed of XML components in different levels and connect them through their relationships extracted from the document structure and content similarity links between them. We then compute a probability score for each component based on the content similarity of the component to the query and propagate the scores in the structure of the network. The result will be a probability score for each component.

These are just a few sample applications for further research. There are possibly many more applications that can utilize the framework.

Study the Characteristics of the Framework

Another line of our future research will be studying different characteristics of the proposed framework. In our framework, computing the scores is an iterative process which continues until convergence. Although the three applications that we have studied in this thesis show the feasibility of doing the iterative process in real time in practice, but we should further study the convergence properties of the framework. We can study the convergence patterns of nodes, the number of iterations needed for convergence and possible ways of speeding up the computation of scores.

Further Generalization of the Framework

In the proposed framework, we currently assume that all the nodes in the network are of the same type. But we can also think of other situations where we have more than one type of node connected in the network and we want to compute different kinds of quality scores for the nodes. For example, we can think of a network composed of terms, topics and documents connected through different kinds of links and we want to score each node. We can further study how to generalize the framework to cover these situations.

References

[1] Review of "trec - experiment and evaluation in information retrieval" by ellen m. voorhees, donna k. harman (eds.), the mit press, cambridge, ma, 2005. *Inf. Process. Manage.*, 43(1):285–287, 2007. Reviewer-Gheorghe Muresan.

[2] V.N. Anh and A. Moffat. Melbourne university 2004: Terabyte and web tracks. In *Proceedings of the TREC Conference*, 2004.

[3] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. Formal models for expert finding in enterprise corpora. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 43–50, New York, NY, USA, 2006. ACM.

[4] Krishna Bharat and Monika R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 104–111, New York, NY, USA, 1998. ACM Press.

[5] Krishna Bharat and George A. Mihaila. When experts agree: using non-affiliated experts to rank popular topics. *ACM Trans. Inf. Syst.*, 20(1):47–58, 2002.

[6] Allan Borodin, Gareth O. Roberts, Jeffrey S. Rosenthal, and Panayiotis Tsaparas. Link analysis ranking: algorithms, theory, and experiments. *ACM Trans. Inter. Tech.*, 5(1):231–297, 2005.

[7] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, and S. Rajagopalan. Automatic resource list compilation by analyzing hyperlink structure and associated text. In *Proceedings of the 7th International World Wide Web Conference*, 1998.

[8] S. Chakrabarti, B. Dom, D. Gibson, S.R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Spectral filtering for resource discovery. In *ACM SIGIR workshop on Hypertext Information Retrieval on the Web*, 1998.

[9] Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, volume 27, pages 307–318. ACM Press, 1998.

[10] Paul R. Cohen and Rick Kjeldsen. Information retrieval by constrained spreading activation in semantic networks. *Inf. Process. Manage.*, 23(4):255–268, 1987.

[11] N. Craswell and D. Hawking. Overview of the trec-2002 web track. In *Proceedings of the TREC Conference*, 2002.

[12] N. Craswell and D. Hawking. Overview of the trec-2003 web track. In *Proceedings of the TREC Conference*, 2003.

[13] N. Craswell and D. Hawking. Overview of the trec-2004 web track. In *Proceedings of the TREC Conference*, 2004.

[14] Nick Craswell and Martin Szummer. Random walks on the click graph. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 239–246, New York, NY, USA, 2007. ACM.

[15] Fabio Crestani and Puay Leng Lee. Searching the web by constrained spreading activation. *Inf. Process. Manage.*, 36(4):585–605, 2000.

[16] W. B. Croft, T. J. Lucia, and P. R. Cohen. Retrieving documents by plausible inference: a priliminary study. In *SIGIR '88: Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 481–494, New York, NY, USA, 1988. ACM Press.

[17] W. B. Croft, T. J. Lucia, J. Cringean, and P. Willett. Retrieving documents by plausible inference: an experimental study. *Inf. Process. Manage.*, 25(6):599–614, 1989.

[18] Brian D. Davison. Toward a unification of text and link analysis. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 367–368, New York, NY, USA, 2003. ACM Press.

[19] Hui Fang and ChengXiang Zhai. Probabilistic models for expert finding. In *ECIR*, pages 418–430, 2007.

[20] Martin Franz and J. Scott McCarley. Arabic information retrieval at ibm. In *TREC*, 2002.

[21] Martin Franz, J. Scott McCarley, and Salim Roukos. Ad hoc and multilingual information retrieval at IBM. In *Text REtrieval Conference*, pages 104–115, 1998.

[22] H. P. Frei and D. Stieger. The use of semantic links in hypertext information retrieval. *Inf. Process. Manage.*, 31(1):1–13, 1995.

[23] Mark Edwin Frisse. Searching for information in a hypertext medical handbook. In *HYPERTEXT '87: Proceedings of the ACM conference on Hypertext*, pages 57–66, New York, NY, USA, 1987. ACM.

[24] Norbert Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[25] Pascale Fung and Lo Yuen Yee. An ir approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 17th international conference on Computational linguistics*, pages 414–420, Morristown, NJ, USA, 1998. Association for Computational Linguistics.

[26] Richard Furuta, Catherine Plaisant, and Ben Shneiderman. A spectrum of automatic hypertext constructions. *Hypermedia*, 1(2):179–195, 1989.

[27] E. Garfield. Citation indexes for science. *Science*, 129, 1955.

[28] Lise Getoor and Christopher P. Diehl. Link mining: a survey. *SIGKDD Explor. Newsl.*, 7(2):3–12, 2005.

[29] Zoubin Ghahramani. Learning dynamic Bayesian networks. *Lecture Notes in Computer Science*, 1387:168–197, 1998.

[30] G. Grimmett and D. Stirzaker. Probability and random processes. In *Oxford University Press*, 1989.

[31] C. O. Hartman. *Virtual Muse: Experiments in Computer Poetry (Wesleyan Poetry)*. Wesleyan University Press, 1996.

[32] T. H. Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):784–796, 2003.

[33] Djoerd Hiemstra and Wessel Kraaij. Twenty-one at trec7: Ad-hoc and cross-language track. In *Text REtrieval Conference*, pages 174–185, 1998.

[34] R. A. Hummel and S. W. Zucker. On the foundation of relaxation labeling processes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5:267–287, 1983.

[35] J. Kamps, G. Mishne, and M. de Rijke. Language models for searching in web corpora. In *Proceedings of the TREC Conference*, 2004.

[36] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[37] Oren Kurland and Lillian Lee. Corpus structure, language models, and ad hoc information retrieval. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 194–201, New York, NY, USA, 2004. ACM Press.

[38] Oren Kurland and Lillian Lee. Pagerank without hyperlinks: structural re-ranking using links induced by language models. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 306–313, New York, NY, USA, 2005. ACM Press.

[39] John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119, New York, NY, USA, 2001. ACM Press.

[40] R. Larson. Bibliometrics of the world wide web: An exploratory analysis of the intellectual structure of cyberspace. In *Annual Meeting of the American Society of Information Science*, 1996.

[41] Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 120–127, New York, NY, USA, 2001. ACM.

[42] R. Lempel and S. Moran. The stochastic approach for link-structure analysis (salsa) and the tkc effect. *Comput. Netw.*, 33(1-6):387–401, 2000.

[43] Xiaoyong Liu and W. Bruce Croft. Cluster-based retrieval using language models. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 186–193, New York, NY, USA, 2004. ACM Press.

[44] Massimo Marchiori. The quest for correct information on the Web: Hyper search engines. *Computer Networks and ISDN Systems*, 29(8–13):1225–1236, 1997.

[45] Hiroshi Masuichi, Raymond Flournoy, Stefan Kaufmann, and Stanley Peters. A bootstrapping method for extracting bilingual text pairs. In *Proceedings of the 18th conference on Computational linguistics*, pages 1066–1070, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

[46] Fabien Mathieu and Mohamed Bouklit. The effect of the back button is a random walk: Application for pagerank. In *Proceedings of the 13th International World Wide Web Conference*, 2004.

[47] S. Meyn. *Control Techniques for Complex Networks*. Cambridge University Press, 2007.

[48] S. Meyn and R. Tweedie. *Markov Chains and Stochastic Stability*. Springer-Verlag, 1993.

[49] David R. H. Miller, Tim Leek, and Richard M. Schwartz. A hidden markov model information retrieval system. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 214–221, New York, NY, USA, 1999. ACM Press.

[50] Dharmendra S. Modha and W. Scott Spangler. Clustering hypertext with applications to web searching. In *HYPERTEXT '00: Proceedings of the eleventh ACM on Hypertext and hypermedia*, pages 143–152, New York, NY, USA, 2000. ACM Press.

[51] Dragos Stefan Munteanu and Daniel Marcu. Improving machine translation performance by exploiting non-parallel corpora. *Comput. Linguist.*, 31(4):477–504, 2005.

[52] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Stable algorithms for link analysis. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 258–266, New York, NY, USA, 2001. ACM Press.

[53] Douglas W. Oard and Anne R. Diekema. Cross-language information retrieval. In *Annual Review of Information Science and Technology*, volume 33, pages 223 – 256, 1998.

[54] Douglas W. Oard and Fredric C. Gey. The trec 2002 arabic/english clir track. In *TREC*, 2002.

[55] Paul Ogilvie and Jamie Callan. Combining document representations for known-item search. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 143–150, New York, NY, USA, 2003. ACM.

[56] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library, 1998.

[57] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[58] E. Picchi and C. Peters. Cross language information retrieval: A system for comparable corpus querying. In *Workshop on Cross-Linguistic Information Retrieval, SIGIR'96*, pages 24 – 33, 1996.

[59] Peter Pirolli, James Pitkow, and Ramana Rao. Silk from a sow's ear: Extracting usable structures from the web. In *Proc. ACM Conf. Human Factors in Computing Systems, CHI*. ACM Press, 1996.

[60] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, New York, NY, USA, 1998. ACM.

[61] Tao Qin, Tie-Yan Liu, Xu-Dong Zhang, Zheng Chen, and Wei-Ying Ma. A study of relevance propagation for web search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 408–415, New York, NY, USA, 2005. ACM Press.

[62] Prabhakar Raghavan Rajeev Motwani. *Randomized Algorithms*. Cambridge University Press, 1995.

[63] Reinhard Rapp. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 320–322, Morristown, NJ, USA, 1995. Association for Computational Linguistics.

[64] Mathew Richardson and Pedro Domingos. The intelligent surfer: Probabilistic combination of link and content information in pagerank. In *Advances in Neural Information Processing Systems*, 2002.

[65] Stephen Robertson. Threshold setting and performance optimization in adaptive filtering. *Information Retrieval*, 5(2-3):239–256, 2002.

[66] Stephen E. Robertson and Karen Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, 1976.

[67] J ROCCHIO. Relevance feedback in information retrieval. In *In The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. PrenticeHall, 1971.

[68] Fatiha Sadat, Masatoshi Yoshikawa, and Shunsuke Uemura. Bilingual terminology acquisition from comparable corpora and phrasal translation to cross-language information retrieval. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 141–144, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[69] Gerard Salton. Associative document retrieval techniques using bibliographic information. *J. ACM*, 10(4):440–457, 1963.

[70] Gerard Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[71] Gerard Salton and Chris Buckley. On the use of spreading activation methods in automatic information retrieval. Technical report, Ithaca, NY, USA, 1988.

[72] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.

[73] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[74] Gerard Salton, C. S. Yang, and C. T. Yu. A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science*, 26(1):33–44, 1975.

[75] Jacques Savoy. Bayesian inference networks and spreading activation in hypertext systems. *Inf. Process. Manage.*, 28(3):389–406, 1992.

[76] Jacques Savoy. Ranking schemes in hybrid boolean systems: a new approach. *J. Am. Soc. Inf. Sci.*, 48(3):235–253, 1997.

[77] Jacques Savoy and Yves Rasolofo. Report on the trec 11 experiment: Arabic, named page and topic distillation searches. In *TREC*, 2002.

[78] Azadeh Shakery and Chengxiang Zhai. Relevance propagation for topic distillation uiuc trec 2003 web track experiments. In *Proceedings of the TREC Conference*, 2003.

[79] P. Sheridan, J. Ballerini, and P. Schauble. Building a large multilingual test collection from comparable news documents. In *G. Grefenstette, editor, Cross-Language Information Retrieval*, Boston, Massachusetts, 1998. Kluwer Academic Publishers.

[80] Luo Si and Jamie Callan. Modeling search engine effectiveness for federated search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 83–90, New York, NY, USA, 2005. ACM.

[81] Amit Singhal, Chris Buckley, and Mandar Mitra. Pivoted document length normalization. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 21–29, New York, NY, USA, 1996. ACM.

[82] H. Small. Co-citation in the scientific literature: a new measure of the relationship between two documents. *The American Society of Information Science*, 24, 1973.

[83] R. Song, J. R. Wen, S. M. Shi, T. Y. Xin, G. M. abd Liu, T. Qin, X. Zheng, J. Y. Zhang, G. R. Xue, and W. Y. Ma. Microsoft research asia at web track and terabyte track of trec 2004. In *Proceedings of the TREC Conference*, 2004.

[84] Marcin Sydow. Random surfer with back step. In *Proceedings of the 13th International World Wide Web Conference*, 2004.

[85] Tao Tao, Xuanhui Wang, Qiaozhu Mei, and ChengXiang Zhai. Language model information retrieval with document expansion. In *HLT-NAACL*, 2006.

[86] Tao Tao and ChengXiang Zhai. Mining comparable bilingual text corpora for cross-language information integration. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 691–696, New York, NY, USA, 2005. ACM Press.

[87] Anastasios Tombros. The effectiveness of query-based hierarchic clustering of documents for information retrieval. Technical report, Glasgow : University of Glasgow, 2002.

[88] T. Tomiyama, K. Karoji, T. Kondo, Y. Kakuta, and T. Takagi. Meiji university web, novelty and genomics track experiments. In *Proceedings of the TREC Conference*, 2004.

[89] Stephen Tomlinson. Experiments in named page finding and arabic retrieval with hummingbird searchservertm at trec 2002. In *TREC*, 2002.

[90] Howard Turtle and W. Bruce Croft. Evaluation of an inference network-based retrieval model. *ACM Trans. Inf. Syst.*, 9(3):187–222, 1991.

[91] C. J. van Rijsbergen. A theoretical basis for the use of co-occurrence data in information retrieval. *Journal of Documentation*, 33:106–119, June 1977.

[92] C. J. Van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979.

[93] C. J. van Rijsbergen. A non-classical logic for information retrieval. *The Computer Journal*, 29(6):481–485, 1986.

[94] Ellen M. Voorhees. The cluster hypothesis revisited. In *SIGIR '85: Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 188–196, New York, NY, USA, 1985. ACM.

[95] Ross Wilkinson and Alan F. Smeaton. Automatic link generation. *ACM Comput. Surv.*, page 27, 1999.

[96] Peter Willett. Recent trends in hierarchic document clustering: a critical review. *Inf. Process. Manage.*, 24(5):577–597, 1988.

[97] S. K. M. Wong and Y. Y. Yao. On modeling information retrieval with probabilistic inference. *ACM Trans. Inf. Syst.*, 13(1):38–68, 1995.

[98] Jinxi Xu and W. Bruce Croft. Query expansion using local and global document analysis. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 4–11, New York, NY, USA, 1996. ACM.

[99] F. Zanettin. Bilingual comparable corpora and the training of translators. In *Laviosa, Sara. (ed.) META, 43:4, Special Issue. The corpus-based approach: a new paradigm in translation studies*, pages 616–630, 1998.

[100] H. Zaragoza, N. Craswell, M. Taylor, S. Saria, and S. Robertson. Microsoft cambridge at trec 13: Web and hard tracks. In *Proceedings of the TREC Conference*, 2004.

[101] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410, New York, NY, USA, 2001. ACM Press.

[102] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410, New York, NY, USA, 2001. ACM Press.

[103] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334–342, New York, NY, USA, 2001. ACM Press.

[104] ChengXiang Zhai and John Lafferty. Two-stage language models for information retrieval. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56, New York, NY, USA, 2002. ACM.

[105] Z. Zhou, Y. Guo, B. Wang, X. Cheng, H. Xu, and G. Zhang. Trec 2004 web track experiments at cas-ict. In *Proceedings of the TREC Conference*, 2004.

Author's Biography

Azadeh Shakery was born in Manchester, United kingdom in 1978. She received her Bachelors and Masters degrees from Sharif University of Technology in 2000 and 2002 respectively. She joined the computer science program in University of Illinois at Urbana-Champaign in August 2002 and completed her Ph.D. in May 2008. Her main research interests are text information management and information retrieval, databases and data mining.